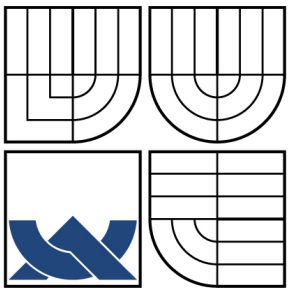
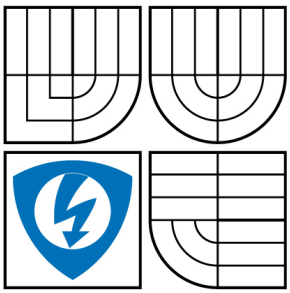


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

VZDÁLENÝ SBĚR DAT S POMOCÍ GSM MODEMŮ NA BÁZI PLATFORMY OPEN AT

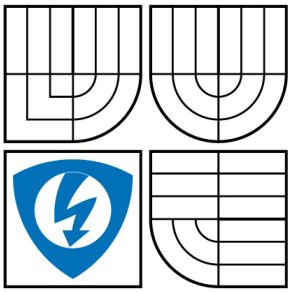
REMOTE DATA ACQUISITION WITH THE HELP OF GSM MODEMS BASED ON OPEN AT PLATFORM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — PAVEL OTOUPALÍK
AUTHOR

VEDOUCÍ PRÁCE — Ing. PAVEL ŠILHAVÝ, Ph.D.
SUPERVISOR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student: Otoupalík Pavel ***ID:*** 78437
Ročník: 3 ***Akademický rok:*** 2007/2008

NÁZEV TÉMATU:

Vzdálený sběr dat s pomocí GSM modemů na bázi platformy Open AT

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte vlastnosti platformy "Open AT", možnosti tvorby uživatelských AT příkazů a autonomní aplikace. Realizujte aplikaci umožňující monitorovat teplotu. Vytvořte jednoduchý modul převodu teploty na kmitočet, který bude prostřednictvím vstup/výstupních pinů připojen k GSM modemu. Monitorovaná data, zejména aktuální čas, datum a teplota, budou prostřednictvím GPRS předávány na FTP server. Získaná měření budou vhodnou formou průběžně zobrazována na Webové stránce. Parametry sběru informací z GSM modemu, zejména četnost a související přesnost měření, bude možno měnit pomocí terminálu.

DOPORUČENÁ LITERATURA:

[1] Herout, P. Učebnice jazyka C - 1. díl. Kopp České Budějovice , ISBN 80-7232-220-6.
[2] Herout, P. Učebnice jazyka C - 2.díl. Kopp České Budějovice , ISBN 80-7232-221-4.
[3] Wavecom, Getting started with Open AT [online] Rev. C. Aktualizováno 2004-06-24. Ve formátu PDF. Dostupný z WWW: http://www.wavecom.com

Termín zadání: 11.2.2008 ***Termín odevzdání:*** 4.6.2008

Vedoucí práce: Ing. Pavel Šilhavý, Ph.D.

prof. Ing. Kamil Vrba, CSc.
předseda oborové rady

UPOZORNĚNÍ:

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Pavel Otoupalík
Bytem: Veselka 111/3, 66441, Troubsko
Narozen/a (datum a místo): 25.4.1985, Brno

(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 244/53, 60200 Brno 2
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Ing. Kamil Vrba, CSc.

(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   - ☐ disertační práce
   - ☐ diplomová práce
   - ☑ bakalářská práce

   jiná práce, jejíž druh je specifikován jako .....

(dále jen VŠKP nebo dílo)

Název VŠKP: Vzdálený sběr dat s pomocí GSM modemů na bázi platformy Open AT
Vedoucí/školitel VŠKP: Ing. Pavel Šilhavý, Ph.D.
Ústav: Ústav telekomunikací
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:

- ☑ tištěné formě - počet exemplářů 1
- ☑ elektronické formě - počet exemplářů 1

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.

4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti
   - ☑ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

Anotace

Tato bakalářská práce se zabývá vzdáleným sběrem dat pomocí GSM modemů s podporou platformy Open AT. Výsledkem práce je realizovaný modul, který slouží jako převodník teploty na kmitočet. Tento modul je propojen s programovatelným GSM/GPRS modemem Wavecom Fastrack M1306B a společně jsou určeny k měření teploty od 0 do 40 °C. S podporou platformy Open AT a programovacího jazyka C byly vytvořeny dvě aplikace. První aplikace naměřenou teplotu společně s datem a časem měření posílá na vzdálený FTP server přes GPRS prostřednictvím GSM sítě. Výsledky měření jsou zobrazovány na webové stránce ve formě tabulky a grafu. Druhá aplikace slouží k posílání naměřené teploty ve formě SMS zprávy jako odpověď na zprávu příchozí.

Klíčová slova: Open AT, měření teploty, GSM, GPRS, vzdálený sběr dat, programovací jazyk C, AT příkaz, časovač 555, modem Fastrack M1306B

Abstract

This bachelor's thesis deals with remote data acquisition with the assistance of GSM modems based on Open AT platform. The result of thesis is the realized module which servers as a converter temperature to frequency. This module is connected with Wavecom Fastrack M1306B programmable GSM/GPRS modem. They are intended for temperature measurement from 0 to 40 °C together. With the assistance Open AT platform and C programming language were created two applications. The first application sends data as temperature, date and time on remote FTP server via GSM network as GPRS data. The results of measurement are displayed on website in form of the table and graph. The second application sends a temperature measurement data via SMS message as reply to SMS message.

Keywords: Open AT, temperature measurement, GSM, GPRS, remote data acquisition, C programming language, AT command, 555 timer, Fastrack M1306B modem

OTOUPALÍK, P. *Vzdálený sběr dat s pomocí GSM modemů na bázi platformy Open AT*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 67 s. Vedoucí bakalářské práce Ing. Pavel Šilhavý, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Vzdálený sběr dat s pomocí GSM modemů na bázi platformy Open AT“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne …… ……..

(podpis autora)

OBSAH

Seznam obrázků

Seznam tabulek

Seznam zkratek

ADL	Application Development Layer
API	Application Programming Interface
APN	Access Point Name
ASCII	American Standard Code for Information Interchange
CMOS	Complementary Metal–Oxide–Semiconducto
DNS	Domain Name System
eDLib	IP Connectivity Development Library
FTP	File Transfer Protocol
GPIO	General Purpose Input/Output
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile communications
NTC	Negative Temperature Coefficient
POP3	Post Office Protocol version 3
PPP	Point-to-Point Protocol
RAM	Random Access Memory
SIM	Subscriber Identity Module
SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

Úvod

Pod pojmem sběru dat si můžeme představit shromažďování určitých informací z jednoho nebo více míst za účelem jejich centralizace, přenosu nebo zpracování, které může být ruční, mechanické, poloautomatické nebo zcela automatické. Dálkové zpracování dat vzniklo spojením výpočetní techniky s telekomunikační technikou do jednoho komplexu.

Sběr dat se realizuje pomocí technických prostředků, které jsou připojeny k nějakému komunikačnímu zařízení (PC, mobil, apod.). Tyto komunikační zařízení zajistí přenos dat do předem definovaného cíle, kde jsou následně určitým způsobem zpracovány, uloženy nebo zobrazeny.

Tato bakalářská práce se zabývá sběru dat pomocí GSM/GPRS modemu Fastrack M1306B od francouzské firmy Wavecom. Modem obsahuje integrovaný mikrokontrolér s podporou platformy Open AT, což nám umožňuje vytvářet vlastní aplikace přímo fungující na modemu. Platforma Open AT nabízí podporu velké řady síťových protokolů, možnost definovat vlastní funkce AT příkazů, a také možnost využívat služeb a hardwarových prostředků modemu v programovacím jazyce C. V této práci je využito vývojového prostředí programu Microsoft Visual C++.

Obsahem této bakalářské práce je navrhnout modul pro převod teploty na kmitočet, který bude připojen k modemu a bude s ním komunikovat pomocí programovatelných vstupně/výstupních pinů s logickými úrovněmi CMOS. Naměřená teplota spolu s dobou měření mohou tak být pomocí modemu například posílány v určitých časových intervalech přes GPRS na vzdálený FTP server nebo pomocí textových SMS zpráv posílány prostřednictvím GSM sítě na mobilní telefon.

1. Návrh modulu pro měření teploty

Navržený modul provádí měření teploty tak, že generuje obdélníkový signál, jehož frekvence je závislá na teplotě. Obdélníkový signál je generován pomocí časovače 555 zapojeného jako astabilní klopný obvod. V zapojení je využito termistoru, který ovlivňuje kmitočet generovaného obdélníkového signálu z výstupu tohoto časovače. Signál je snímán modemem s programovatelným vstupně/výstupním portem s logickými úrovněmi CMOS. Modem provede měření teploty tak, že zjistí délku doby, kdy je na nastaveném vstupním pinu jeho portu logická úroveň H. Schéma zapojení modulu je uvedeno v příloze A.1.

1.1. Návrh obvodu napájení

Modul obsahuje stabilizátor napětí s integrovaným obvodem TL317CD, který je zapojený podle doporučení udávané výrobcem v katalogovém listu [8]. Filtrační kondenzátory C_1 a C_2 zlepšují stabilitu a odezvu na skokovou změnu vstupního napětí nebo velikosti zátěže. Kondenzátor C_3 zlepšuje potlačení zvlnění na výstupu. Při použití kondenzátoru C_3 musí být zapojení doplněno o diodu D_2, která zajistí vybití kondenzátoru C_3 při zkratu na výstupu a vypnutí zdroje. Dioda D_1 chrání stabilizátor před zničením zpětným proudem.

Poměr odporů rezistorů R_2 a R_1 nám definuje výstupní napětí stabilizátoru. Obvod nastaví na výstupu takové napětí, aby rozdíl napětí mezi vývody 1 a 3 byl $U_{REF} = 1{,}25$ V. Hodnota odporu rezistoru R_1 je zvolena podle doporučení výrobce 470 Ω, aby byl zajištěn minimální odběr proudu 2,5 mA.

Modem, ke kterému je modul připojen, je schopen rozpoznat logickou úroveň H na svém GPIO portu, je-li na něm připojeno napětí od 2,1 V do 3 V. Minimální napětí pro napájení integrovaných obvodů CMOS, které jsou v modulu použity, je 3 V. Proto je stabilizátor napětí navržen tak, aby na jeho výstupu bylo napětí $U_{OUT} = 3$ V.

Pro výstupní napětí stabilizátoru platí vztah [8]:

$$U_{\mathrm{OUT}} = U_{\mathrm{REF}} \cdot \left(1 + \frac{R_2}{R_1}\right) . \qquad (1.1)$$

Pro výpočet hodnoty odporu R_2, je-li $R_1 = 470\ \Omega$ můžeme psát

$$R_2 = \left(\frac{U_{\mathrm{OUT}}}{U_{\mathrm{REF}}} - 1\right) \cdot R_1 = \left(\frac{3}{1{,}25} - 1\right) \cdot 470 = 658\,\Omega . \qquad (1.2)$$

Časovač TLC555CDR i integrovaný obvod CD40106BM, což je obvod skládající se ze šesti inventorů vybavených Schmittovými klopnými obvody, jsou typu CMOS. Napájecí napětí 3 V bude podle katalogových údajů [6, 7] dostačující.

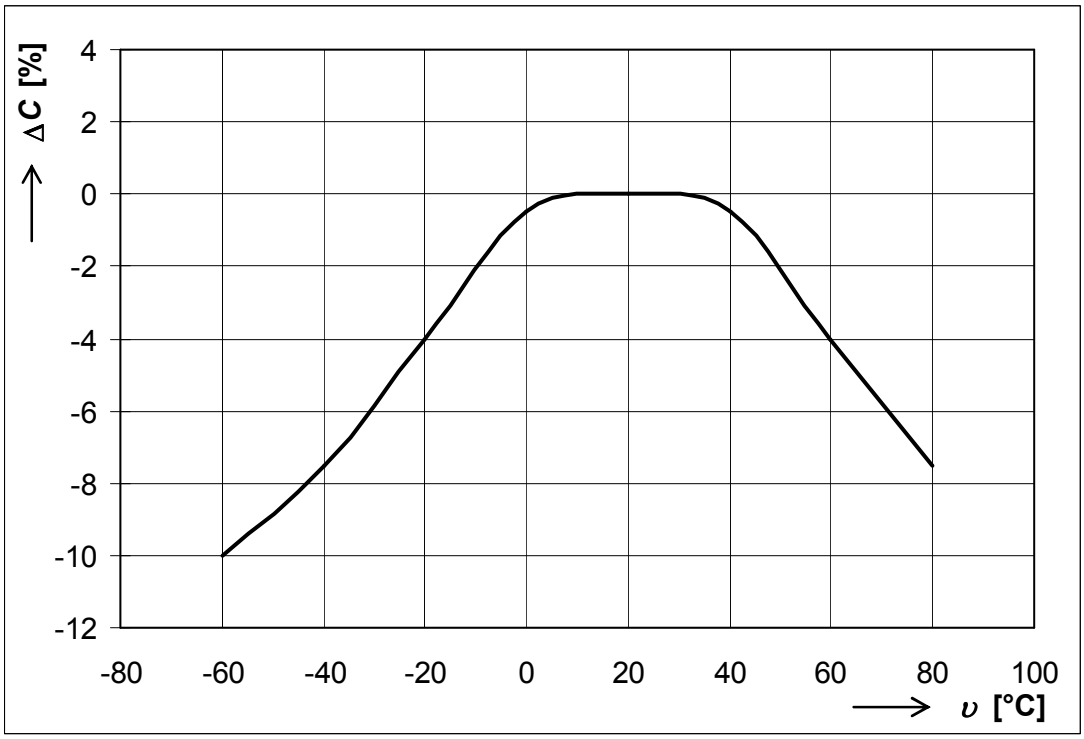
Jako vstupní napětí stabilizátoru lze použít zdroj stejnosměrného napětí *U*cc = 6 až 38 V. Rozdíl mezi vstupním a výstupním napětí stabilizátoru nesmí být podle katalogových listů [8] větší než 35 V. Vstupní filtrační kondenzátor C_1 je dimenzován pro napětí 50 V.

1.2. Návrh převodníku teploty na frekvenci

Časovač TLC555CD je zapojen jako astabilní klopný obvod. Měření teploty je provedeno pomocí termistoru K164NK100, který je typu NTC. To znamená, že má negativní teplotní koeficient a jeho velikost odporu klesá ze vrůstající teplotou. Jeho jmenovitý odpor při teplotě 25 °C je 100 kΩ.

Na výstupu časovače je generován obdélníkový signál, jehož perioda je dána dobou nabíjení a vybíjení kondenzátorů C_5 a C_6. Jedná se o keramické kondenzátory s dielektrikem X5R, které by mělo mít podle katalogových údajů [1] stabilní velikost kapacity od 0 do 40 °C (viz. obr. 1.1).

Kondenzátory C_5 a C_6 se nabijí přes odpor rezistoru R_5 a odpor termistoru TER_1 a vybíjí pouze přes odpor termistor TER_1. T_H nám udává délku doby pro logickou úroveň H a T_L délku dobu pro logickou úroveň L.



Obr. 1.1: Závislost změny kapacity keramických kondenzátorů s dielektrikem X5R na teplotě

Pro výpočet délek dob T_H a T_L platí:

$$T_H = (R_5 + R_{TER1}) \cdot (C_5 + C_6) \cdot \ln 2 \ , \qquad (1.3)$$

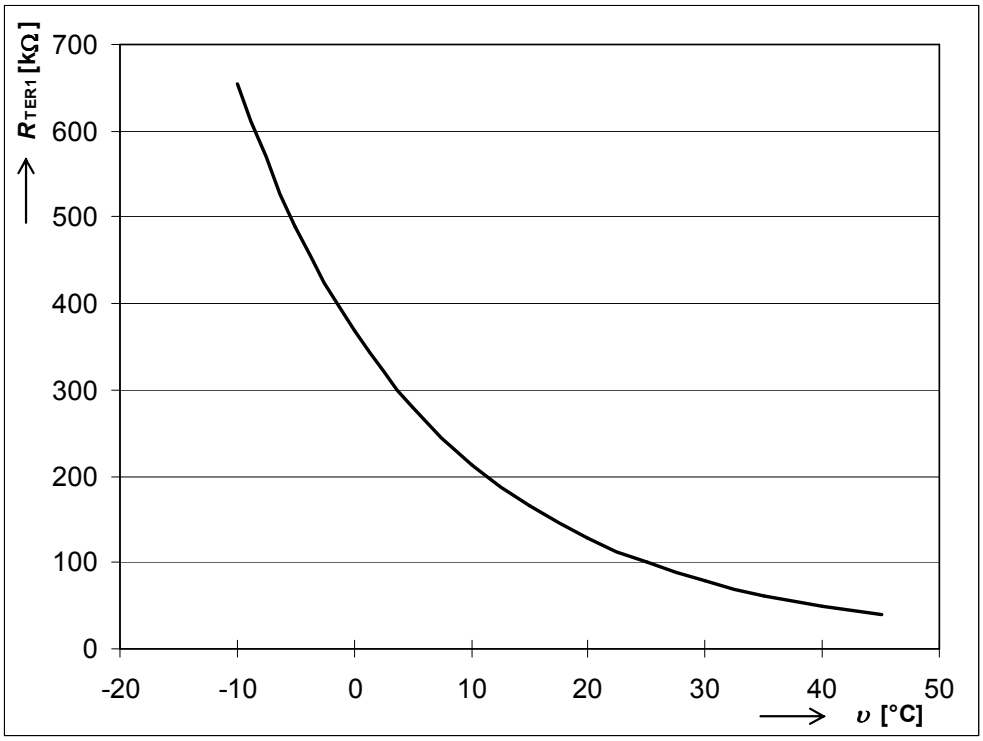
$$T_L = R_{TER1} \cdot (C_5 + C_6) \cdot \ln 2 \ . \qquad (1.4)$$

Celková perioda obdélníkového signálu je dána součtem těchto dob $T = T_H + T_L$. Měření teploty je realizováno pomocí délky doby T_H.

Modem bude snímat stav na svém GPIO portu, na kterém je připojen výstup časovače 555 každých 18,5 ms, což je nejkratší možná doba nastavení časovače použitého modemu FASTRACK M1306B [12]. Frekvence obdélníkového signálu je vypočtena tak, aby přesnost měření byla $\Delta\upsilon$ = ± 0,25 °C pro teplotu pohybující se kolem 25 °C. Přesnost měření je různá pro rozdílné teploty z důvodu nelineární charakteristiky závislosti odporu termistoru TER_1 na teplotě (viz obr. 1.2). Doba pulsů z výstupu časovače 555 je proto tak dlouhá, aby byla zachována výše zmiňovaná přesnost.

Tab. 1.1: Výpočet parametrů obdélníkového signálu z výstupu časovač 555

Teplota υ	R_{TER1}	T_H	T_L	T	f
[°C]	[kΩ]	[s]	[s]	[s]	[Hz]
-10	655	9,19	9,16	18,36	0,05
-5	489	6,87	6,84	13,71	0,07
0	368	5,18	5,15	10,33	0,10
5	280	3,94	3,91	7,86	0,13
10	214	3,02	2,99	6,02	0,17
15	165	2,34	2,31	4,65	0,22
20	128	1,83	1,80	3,62	0,28
25	100	1,43	1,40	2,83	0,35
30	78	1,13	1,10	2,23	0,45
35	62	0,90	0,87	1,76	0,57
40	49	0,72	0,69	1,40	0,71
45	39	0,58	0,55	1,13	0,89
50	31	0,47	0,44	0,91	1,10
pro $C_5 + C_6 = 20$ μF a $R_5 = 2{,}2$ kΩ					



Obr. 1.2: Závislost odporu termistoru K164NK100 na teplotě

1.3. Rozšíření modulu o přídavné funkce

Modul dále obsahuje tlačítko S_1 a světelnou diodu LED_1. Podle nastavení zkratovací propojky JP_1 se dá vybrat, které z těchto dvou zařízení bude připojeno na programovatelný vstupně/výstupní pin portu modemu a bude moci být používáno. Rezistor R_9 slouží pro ošetření vstupu inventoru, jehož výstup je dále připojen k světelné diodě, aby zde nemohl nastat neurčitý stav, je-li světelná dioda odpojena pomocí zkratovací propojky od konektoru.

Kvůli zákmitům vznikajících při stisku tlačítka je do obvodu přidán integrační článek skládající se z rezistoru R_4 a kondenzátoru C_4. Výstupní napětí z integračního článku zpracovává inventor vybavený Schmittovým klopným obvodem. Odpor R_3 slouží pro nastavení proudu tekoucím tlačítkem při sepnutém stavu.

Proud světelnou diodou LED1 je určen podle vztahu:

$$I_{\mathrm{LED1}} = \frac{U_{\mathrm{OUT}} - U_{\mathrm{LED1}}}{R_7} = \frac{3-2}{330} = 3{,}03 \text{ mA} \ . \qquad (1.5)$$

Rezistory R_6 a R_8 slouží k ochraně zařízení proti zkratu na výstupu nebo připojení nevhodného napětí. Kondenzátor C_8 slouží jako blokovací kondenzátor, který má v obvodu za úkol zásobovat blokovaný obvod elektrickým proudem při rychlých změnách jeho odběru.

Integrovaný obvod CD40106BM, který obsahuje šestici inventorů vybavené Schmittovými klopnými obvody, slouží k dosažení vysoké strmosti hran při stisknutí tlačítka a pro definování úrovní CMOS.

Tab. 1.2: Zapojení vývodů konektoru CON_1

Vývod	Funkce
1	GND
2	Tlačítko/LED (podle nastavení zkratovací propojky)
3	Výstup z časovače 555
4	U_{CC}

2. Platforma Open AT

Platforma Open AT nabízí uživateli možnost definovat funkce vlastních AT příkazů. To umožňuje vytvářet aplikace pro řízení a kontrolu bezdrátových modemů v sítí GSM a GPRS. Vlastní aplikaci je možno vytvořit v prostředí programu Microsoft Visual C++ nebo jiného vývojového prostředí jazyka C. Chování aplikace můžeme pozorovat v programu Terminal Emulator, či programu Target Monitoring Tool od firmy Wavecom, nebo také v programu HyperTerminal, který je součástí operačního systému Windows.

Pomocí aplikací vytvořené v prostředí Open AT můžeme využít z některých komunikačních protokolů POP3, SMTP k přijímání a odesílání e-mailů nebo pomocí protokolu FTP odesílat data na vzdálený FTP server. Data mohou být také posílána v síti GSM pomocí textových zpráv SMS.

Jedním z GSM/GPRS modemů, které podporují Open AT je modem Fastrack M1306B, který vyrábí francouzská firma Wavecom. Tento modem může být jako běžný GSM/GPRS modem standardně ovládán počítačem pomocí AT příkazů a lze použít pro GSM pásmo 900 a 1800 MHz. Aby vlastní aplikace mohli být spuštěny na modemu bez nutnosti přídavného mikrokontroleru, modem už obsahuje mikrokontroler integrovaný. Modem FASTRACK umožňuje vložit vlastní softwarovou aplikaci přímo do své paměti. Modem obsahuje programovatelný port se dvěma vstupně/výstupními piny GPIO s logickými úrovněmi CMOS. Modem je připojen k počítači přes sériové rozhraní RS232 a je jím řízen pomocí AT příkazů. K napájení modemu můžeme použit zdroj stejnosměrného napětí od 5,5 do 32 V.

Tab. 2.1: Úrovně CMOS GPIO pinů modemu Fastrack M1306B

Parametr	Min	Max
U_{IL}	-0,5 V	0,8 V
U_{IH}	2,1V	3,0 V
U_{OL}	0 V	0,2 V
U_{OH}	2,6 V	2,8 V
I_{OL}	-2 mA	
I_{OH}	2 mA	

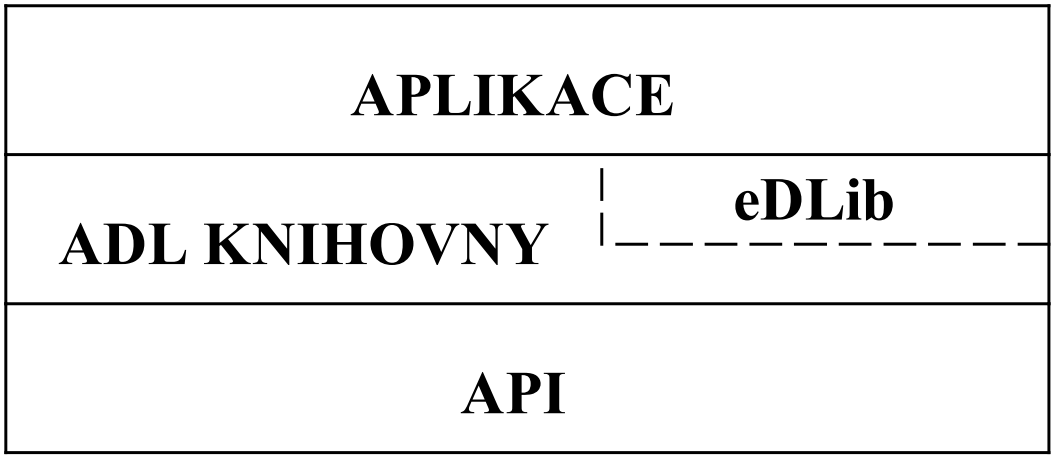
Tab. 2.2: Nastavení sériového portu RS232 pro komunikaci s modem FASTRACK M1306B při použití programu HyperTerminal nebo Terminal Emulator

Parametr	Hodnota
Bitů za sekundu	115 200 b/s
Datové bity	8
Parita	Žádná
Počet stop-bitů	1
Řízení toku	Hardware

2.1. Architektura Open AT

Z programátorského hlediska lze platformu Open AT rozdělit do tří vrstev.

- Programátorem vytvořená aplikace
- Knihovny funkcí
- Aplikační uživatelské rozhraní



Obr. 2.1: Architektura Open AT

2.1.1. Aplikační uživatelské rozhraní API

Jedná o soubor procedur, funkcí či tříd knihoven, kterých programátor využívá. API určuje, jakým způsobem se funkce knihovny mají volat ze zdrojového kódu programu.

2.1.2. ADL knihovny

Open AT využívá vlastní knihovny funkcí, zvané ADL. Tato zkratka vychází z anglické definice Application Development Layer (vrstva pro vývoj aplikací). ADL knihovny jsou navrženy a používány pro zjednodušení programování Open AT aplikací. Chceme-li využit z některou z funkcí z knihoven ADL, musíme do programu importovat hlavičkový soubor (.h) dané ADL knihovny.

V Open AT aplikaci můžeme využívat funkce těchto ADL knihoven:

- adl_at.h - obsahuje funkce pro zadávání AT příkazů
- adl_TimerHandler.h - obsahuje funkce pro práci s časovači
- adl_memory.h - obsahuje funkce pro práci s pamětí RAM
- adl_flash.h - obsahuje funkce pro práci s pamětí FLASH
- adl_gpio.h - obsahuje funkce pro obsluhu portů modemu
- adl_sim.h - obsahuje funkce pro služby SIM karty
- adl_sms.h - obsahuje funkce pro obsluhu SMS zpráv
- adl_call.h - obsahuje funkce pro obsluhu hovorů
- adl_gprs.h - obsahuje funkce pro služby GPRS
- adl_str.h - obsahuje funkce pro obsluhu AT řetězců
- adl_gps.h - obsahuje funkce pro služby GPS

2.1.3. EDLib knihovny

Má-li být v Open AT aplikaci využito z některých internetových komunikačních protokolů (např. POP3, SMTP, FTP, TCP atd.), musí být v programu využito knihoven Open AT IP Connectivity Library označené jako eDLib. Tyto knihovny musí být začleněny do programu pomocí skriptu:
"**-otherlib /TCPIP/v3.00 -inc /TCPIP/v3.00/inc**" (více v kapitole 2.3).

Dále musí být v programu volána funkce **ed_Init()**, která slouží k inicializaci knihovny pro IP spojení. Chceme-li využit v programu z některou z funkcí z knihoven eDLib, musíme opět importovat hlavičkový soubor (.h) dané eDLib knihovny.

V Open AT můžeme využívat funkce těchto eDLib knihoven:

- ed_common.h - obsahuje hlavičkové soubory všech níže popsaných knihoven
- ed_dialup.h -obsahuje funkce pro sestavení spojení

- ed_ppp.h -obsahuje funkce pro využití protokolu PPP
- ed_dns.h -obsahuje funkce pro využití služeb DNS
- ed_gprs.h -obsahuje funkce pro využití protokolu GPRS
- ed_smtp.h -obsahuje funkce pro využití protokolu SMTP
- ed_pop3.h -obsahuje funkce pro využití protokolu POP3
- ed_ftp.h -obsahuje funkce pro využití protokolu FTP
- ed_socket.h -obsahuje funkce pro využití protokolu TCP
- ed_udpsocket.h -obsahuje funkce pro využití protokolu UDP

2.1.4. Aplikace v Open AT

Programátorem vytvořená aplikace v některém vývojovém prostředí jazyka C s využitím ADL a eDLib knihoven na bázi platformy Open AT. Tyto aplikace jsou pak přímo nahrány do paměti modemu a mohou využívat jeho funkce a technické prostředky.

2.2. Datové typy v Open AT

Platforma Open-AT má kromě standardních datových typů definované taky vlastní datové typy, které odpovídají standardním datovým typům používaných v klasickém C++.

Tab. 2.3: Přehled datových typu v Open AT

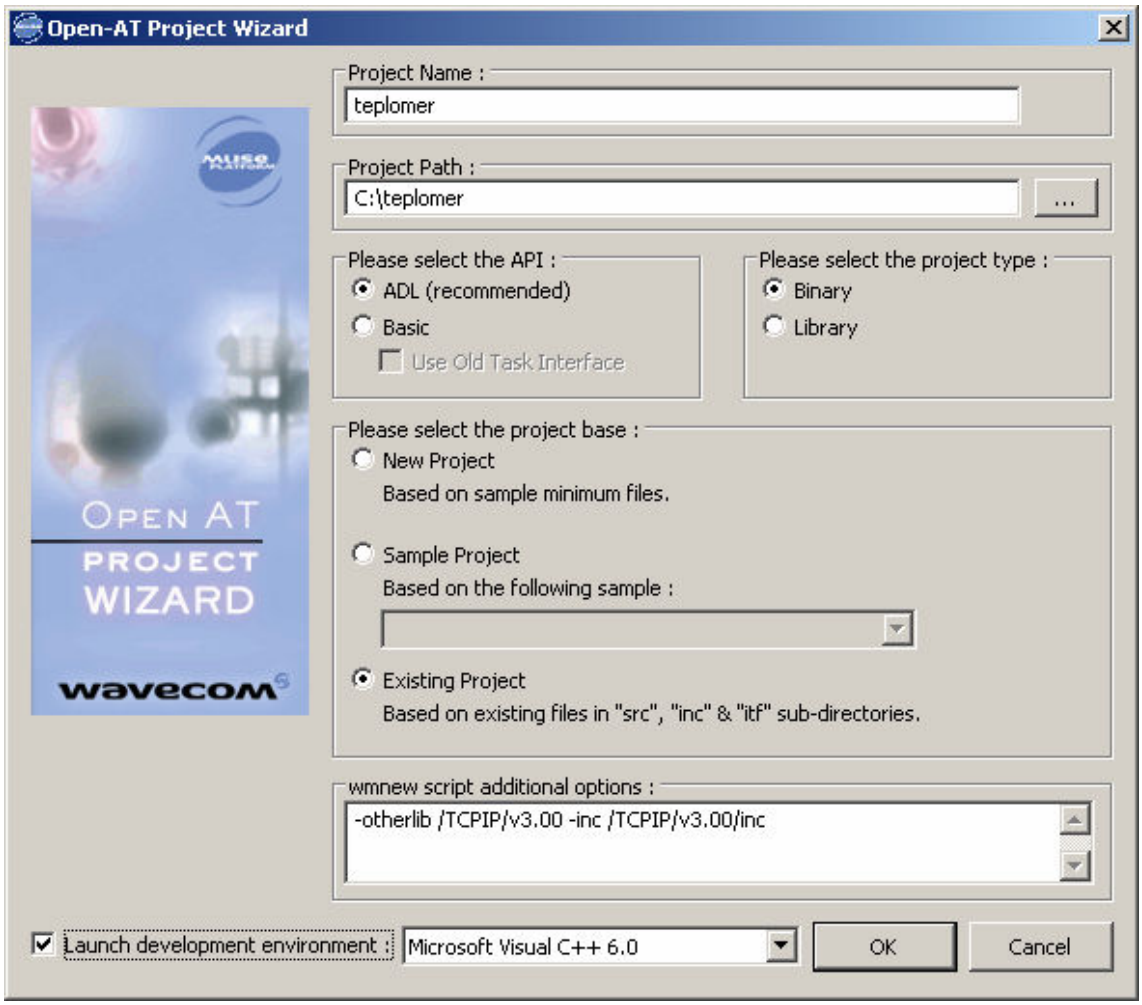
Open-AT Typ	C++ Typ	Délka	Rozmezí
u8	unsigned char	8 bitů	0 až 255
s8	signed char	8 bitů	-128 až 127
ascii	char	8 bitů	-128 až 127
u16	unsigned short	16 bitů	0 až 65 535
s16	short	16 bitů	-32 768 až 32 767
u32	unsigned int	32 bitů	0 až 2^{32}
s32	Int	32 bitů	-2^{31} až $(2^{31} - 1)$

2.3. Tvorba aplikace v Open AT

Pro vytvoření nové Open AT aplikace nebo k úpravě již existující, je nejlepší využít průvodce „Open AT Project Wizard“, který je součástí balíku „Open AT Suite“ (viz obr. 2.2).

Průvodce umožňuje zvolit zejména tato nastavení:

- Název projektu (*Project name*) a jeho umístění na disku (*Project Path*)
- Výběr aplikačního uživatelského rozhraní (*Please select the API*).
    - Pokud je vybráno doporučené ADL API, může být v programu využito navíc i ADL knihoven.
- Typ projektu (*Please select the project type*)
    - Binary – tvorba aplikace
    - Library – tvorba knihovny
- Volba otevření nebo vytvoření nového projektu (*Please select the project base*)
    - New project – vytvoření nového projektu
    - Sample project – otevření ukázkového projektu
    - Existing project – otevření již existujícího projektu
- Pole pro skript sloužící k začlenění přídavných knihoven jazyka C do programu (*wmnew skript additional options*)
    - -otherlib /TCPIP/v3.00 -inc /TCPIP/v3.00/inc - přidá knihovny eDLib
- Výběr vývojového prostředí (*Launch development Environment*)
    - Microsoft Visual C++ 6.0
    - Eclipse
    - None – pouze dojde k vytvoření projektu



Obr. 2.2: Okno průvodce Open AT Project Wizard

Po vytvoření projektu pomocí průvodce „Open At Project Wizard“ se ve zvoleném pracovním adresáři objeví tato struktura:

- **RTE** - Obsahuje aplikaci vygenerovanou programovacím nástrojem pro podporované jazyky.
- **ARM/OUT** - Obsahuje přeložený kód, který je sestaven ARM kompilátorem a přizpůsoben pro nahrání do modemu
- **GCC/OUT -** Obsahuje přeložený kód, který je sestaven GCC kompilátorem a přizpůsoben pro nahrání do modemu
- **INC -** Obsahuje uživatelský Open AT hlavičkový kód aplikace.
- **SRC -** Obsahuje uživatelský Open AT zdrojový kód aplikace

Při vytváření programu je nutné deklarovat dvě povinné proměnné **wm_apmCustomStack** a **wm_apmCustomStackSize**, které jsou nezbytné pro chod programu. Tyto proměnné slouží k alokaci paměti modemu pro tento program.

Příklad deklarace povinných proměnných [9]:

u32 wm_apmCustomStack [256];

/* Hodnota může být například 256 */

const u16 wm_apmCustomStackSize = sizeof(wm_apmCustomStack);

Tělo hlavního programu se nachází uvnitř funkce **void adl_main (adl_InitType_e InitType) {tělo hlavního programu}.**

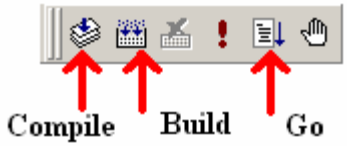
2.4. Spuštění Open AT aplikace

Možnosti jak zprovoznit Open AT aplikaci na modemu jsou dvě.

- Ladící mód (**Debug Mode)**. V této metodě probíhá testování a ladění na základě simulačního modelu zařízení spuštěném na počítači.
- Cílový mód (**Target Mode**) spočívá v přenosu zkompilovaného programu do flash paměti koncového hardwarového zařízení, na kterém je program spuštěn.

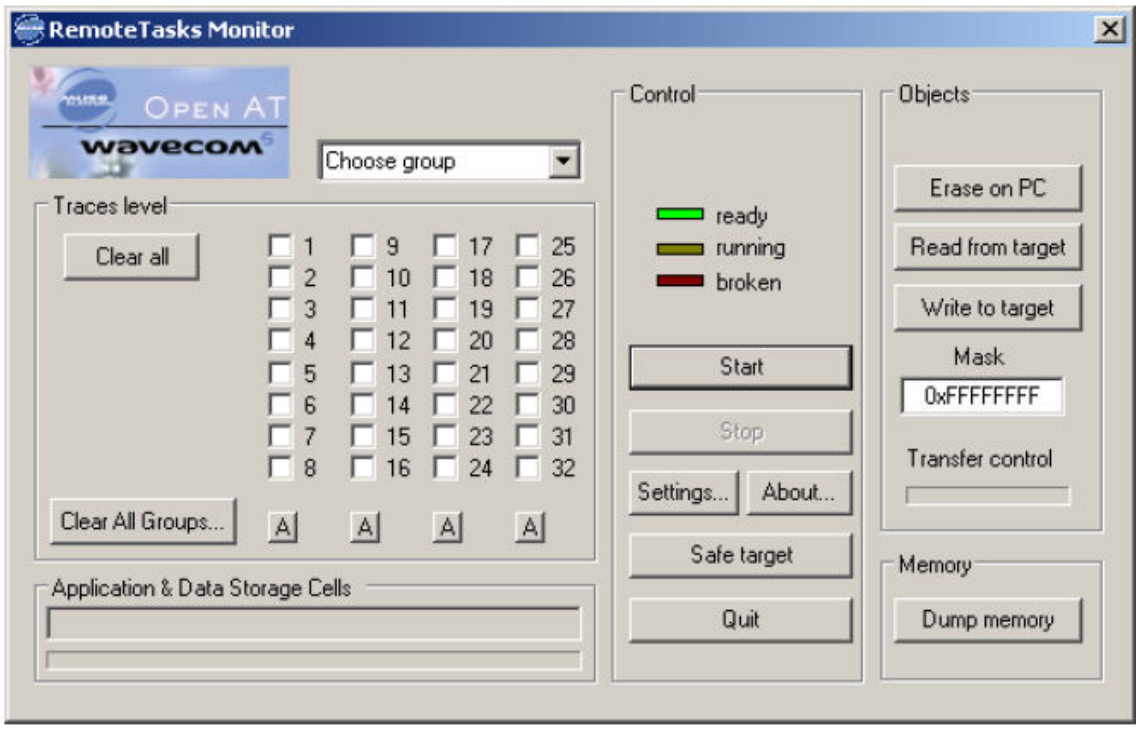
2.4.1. Ladící mód (Debug Mode)

Pro použití první možnosti musí být vybráno ve vývojovém prostředí Microsoft Visual C++ z nabídky (Build -> Set Active Configuration…) položka "Win32 Debug". Aby bylo možno aplikaci spustit, musíme ji nejdříve zkompilovat a sestavit (kliknutím na příkaz "Compile" a "Build"). Do ladícího módu přejdeme po kliknutí na příkaz "Go".



Obr. 2.3: Kompilace a spuštění aplikace v ladícím módu

Dojde ke spuštění programu RemoteTask Monitor, ve kterém spustíme aplikaci tlačítkem “Start“. Tlačítkem “Stop“ pak můžeme aplikaci zastavit. Tlačítko “Quit“ slouží pro ukončení aplikace a návratu do Microsoft Visual C++. Chování spuštěné aplikace můžeme sledovat v programu Terminal Emulator, či programu Target Monitoring Tool.



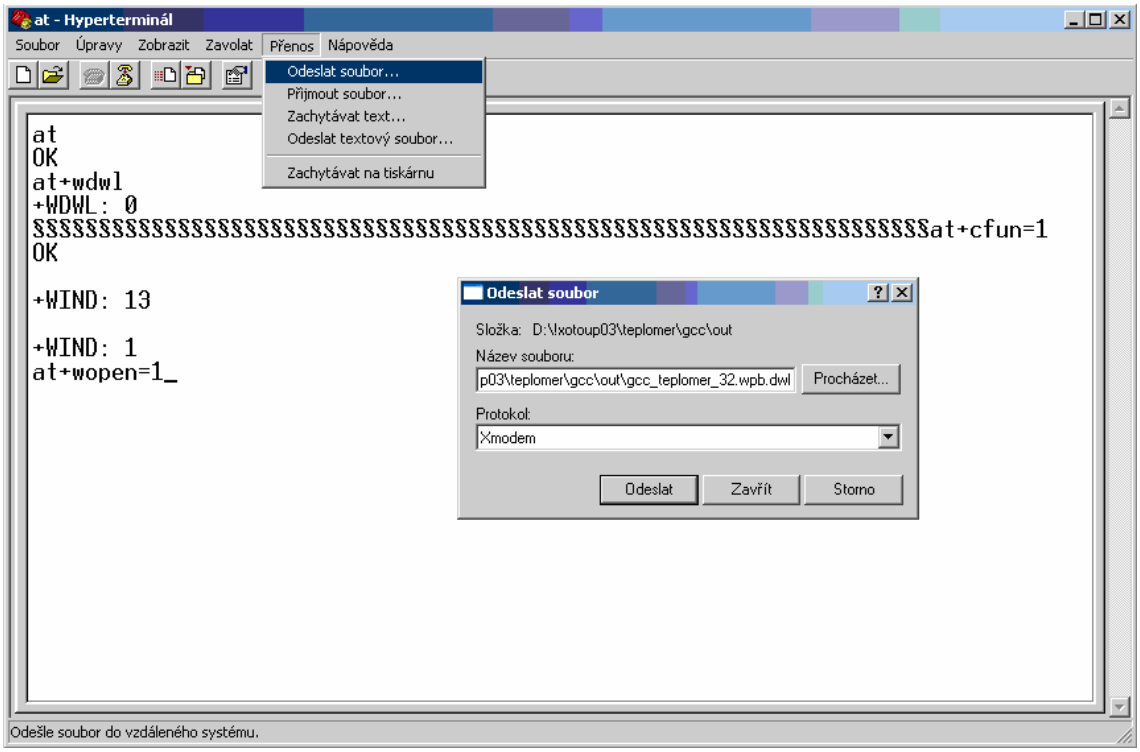
Obr. 2.4: Okno programu RemoteTask Monitor

2.4.2. Cílový mód (Target Mode)

U druhého způsobu, kdy bude zkompilovaná aplikace nahrána a spuštěna přímo na modemu, musí být vybráno ve vývojovém prostředí Microsoft Visual C++ z nabídky (Build -> Set Active Configuration…) položka “Win32 Wismo_Target“. Aplikace se sestaví příkazem “Build“. Podle zvoleného kompilátoru se ve složce ARM/OUT nebo GCC/OUT vytvoří zkompilovaná aplikace. Soubor který je určen pro nahrání do paměti modemu je pojmenován “*zvolenykompilator_jmenoprojektu_32.wdl*“ nebo “*zvolenykompilator_jmenoprojektu_32.wpb.wdl*“, který je navíc komprimován.

V programu HyperTerminal pomocí AT příkazu **AT+WDWL** se modem nastaví do download módu a přes protokol XMODEM se zkompilovaná aplikace nahraje do jeho paměti. Po stáhnutí aplikace se po zadání příkazu **AT+CFUN=1** modem resetuje a provede se nová inicializace. Aplikace lze spustit pomocí AT příkazu

AT+WOPEN=1 a zastavit pomocí příkazu **AT+WOPEN=0.** Pro odstranění programu z paměti modemu slouží příkaz **AT+WOPEN=4**. Příkazem **AT+WOPEN=3** se provede vymazání objektů z paměti flash, které byly vytvořeny spuštěnou aplikací. [11]



Obr. 2.5: Nahrání aplikace do paměti modemu pomocí programu Hyperterminal

3. Využití a obsluha modulu pro měření teploty

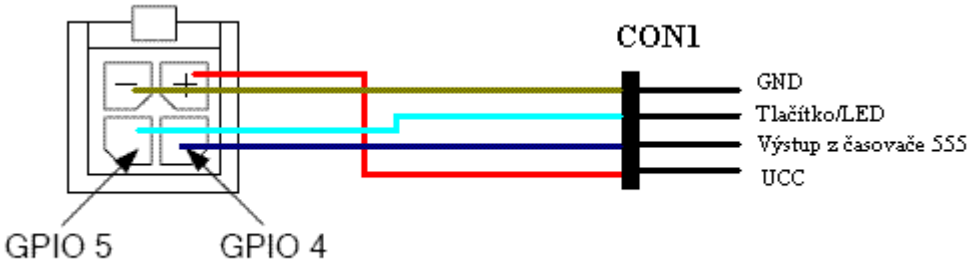
Modul pro měření teploty je připojen k modemu Fastrack M1306B pomocí dvou vstupně/výstupních pinů. Jeden pin, na modemu označeného jako GPIO4, musí být vždy nastaven jako vstupní. Tento pin slouží ke snímání výstupního signálu z modulu pro měření teploty. Z doby pulsu snímaného signálu je pak pomocí aplikace vypočtena měřená teplota.

Druhý pin, označený na modemu jako GPIO5, může být použit jak pro vstup tak i výstup z modemu. Modul obsahuje zkratovací propojku, pomocí které se zvolí, zda se má daný pin chovat jako vstup či výstup. Pokud bude port nastavený jako vstup, můžeme k němu pomocí propojky připojit tlačítko umístěné na desce modulu. Pokud

bude port nastaven jako výstup, lze k němu připojit signalizační LED diodu rovněž umístěné na desce.

Na desce plošného spoje jsou vedle zkratovací propojky umístěny popisky, aby bylo zřejmé, do jaké pozice se má zkratovací propojky zapojit. Nápis Ld1 značí, že k pinu modemu bude připojena LED dioda a pin musí být nastaven tedy jako výstupní. Pokud zkratovací propojku umístíme do pozice Tl1, bude k pinu modemu připojené tlačítko a pin tedy musí být nastaven jako vstupní.

Na modulu je dále umístěn termistor TER1, který je použit v obvodu pro měření teploty. Modul je připojen k modemu Fastrack M1306B pomocí konektoru podle obrázku 3.1.



Obr. 3.1: Připojení modulu pro měření teploty k programovatelným vstupně/výstupním pinům modemu Fastrack M1306B

3.1. Způsob použití modulu pro měření teploty

V tabulce 3.1 jsou uvedeny vypočítané hodnoty délek dob T_H pro logickou úroveň H z výstupu časovače 555, které je zapojen jako astabilní klopný obvod, který mění svoji frekvenci v závislosti na teplotě.

Délka doby T_H bude právě využita pro zjištění měřené teploty. A to tak, že modem Fastrack M1306B bude snímat stav na svém GPIO portu, na kterém je připojen výstup časovače 555 každých 18,5 ms, což je nejkratší možná doba nastavení časovače použitého modemu FASTRACK M1306B [12]. Z počtu zachycených vzorků v době T_H bude program nahraný v paměti modemu schopen vypočítat měřenou teplotu v rozsahu od 0 do 40 °C.

Počet skutečně zachycených vzorků modemem se ale liší od teoreticky vypočítaných vzorků, které by měl modem zachytit. Měřením bylo zjištěno, že počet

zachycených vzorků modemem je přibližně o 17 menší. Proto je proveden přepočet počtu zachycených vzorků jak ukazuje tabulka 3.1.

Tab. 3.1: Přepočet zachycených vzorků modemem při měření teploty

Teplota υ	T_H	Počet zachycených vzorků modemem v době T_H	
°C	s	vypočteno	Odhadnuto
0	5,18	280	263
5	3,94	213	196
10	3,03	164	147
15	2,34	127	110
20	1,83	99	82
25	1,43	75	58
30	1,13	61	44
35	0,90	48	31
40	0,72	39	22

Pro převod zachycených vzorků modemem na teplotu bude v programu využito rovnice získané z logaritmické regrese [4]:

$$y = B + A \ln x \ , \tag{3.1}$$

Konstanty A, B jsou pak dány těmito vztahy [4];

$$B = \frac{n \cdot \sum (\ln x) y - \sum \ln x \cdot \sum y}{n \cdot \sum (\ln x)^2 - (\sum \ln x)^2} \ , \tag{3.2}$$

$$A = \frac{\sum y - B \cdot \sum \ln x}{n} \ . \tag{3.3}$$

Výsledná rovnice pro výpočet teploty je tedy dána vztahem:

$$y = B + A \ln x = -16{,}199 + \ln 90{,}758 \,°\mathrm{C} \ . \tag{3.4}$$

Kde y je teplota ve °C a x je počet zachycených vzorků modemem v době T_H.

4. Aplikace pro měření teploty s přenosem dat na FTP server.

Program slouží pro měření teploty na principu převodu teploty na frekvenci. Pro správnou funkci tohoto programu je potřeba nastavit zkratovací propojku umístěnou na modulu do pozice **Ld1**. Po spuštění programu je z hlavní části programu volán cyklický časovač, nastavený na 18,5 ms. Při každém jeho přetečení se opakovaně čte stav pinu GPIO4, ke kterému je připojen výstup časovače 555. Je-li na vstupu pinu log.úroveň H je provedena inkrementace proměnné **pocet**, jejíž hodnota značí délku pulzu. Je-li na vstupu log. úroveň L dojde k převodu proměnné **pocet** na teplotu. V programu je také testován stav pinu GPIO5, ke kterému je připojena LED dioda, sloužící k signalizaci přenosu dat na FTP server prostřednictvím GPRS. Interval, po kterém se naměřená teplota spolu dobou měření budou posílat na FTP server, lze měnit v terminálu pomocí AT příkazu **AT+FTPTIME**. V programu jsou dále nadefinovány další AT příkazy pro změnu konfiguračních údajů GPRS a FTP připojení. Jejich názvy, příklady použití a stručný popis jsou uvedeny v tabulce 4.1. Všechny údaje, které se dají pomocí těchto AT příkazu měnit, jsou ukládány do flash paměti modemu. To znamená, že v případě ukončení nebo neočekávaného přerušení programu, jsou při opětovném spuštění aplikace tyto údaje nahrány z paměti a nemusí se znovu konfigurovat.

Program získává údaje o čase a datu z modemu. Správné nastavení těchto údajů se dá zjistit za použití příkazu **AT+CCLK?**. Pokud tomu tak není, nastaví se přesný čas a datum pomocí stejného příkazu s parametrem **AT+CCLK="rr/mm/dd, hh:mm:ss"** (např. AT+CCLK="08/05/02,09:00:00").

Pro správnou funkci programu, musí být zkompilovaná aplikace nahrána a spuštěna na modemu v cílovém módu (viz. kap. 2.4.2). Získaná měření jsou zobrazena na webové stránce http://openat-teplota.ic.cz formou tabulky. Na této stránce si lze také zobrazit naměřené teploty během jednoho dne formou grafu. Zdrojový kód webové stránky je psán v HTML a PHP kódu a je uveden v příloze C.

Tab. 4.1: AT příkazy vytvořené v programu

AT příkaz	Příklad použití	Popis
AT+FTPTIME	AT+FTPTIME=20	Nastaví periodické posílání dat na FTP server každých 20 minut
	AT+FTPTIME=?	Vypíše aktuální nastavení jak často jsou data posílána na FTP server
AT+FTPSERV	AT+FTPSERV=“ic.cz“	Nastaví adresu FTP serveru
	AT+FTPSERV=“88.86.103.242“	
	AT+FTPSERV=?	Vypíše adresu aktuálního FTP serveru
AT+FTPUN	AT+FTPUN=“openat-teplota“	Nastaví uživatelské jméno pro přihlášení na FTP server
	AT+FTPUN=?	Zobrazí nastavené uživatelské jméno pro přihlášení na FTP server
AT+FTPPW	AT+FTPPW=“qwert“	Nastaví heslo pro přihlášení na FTP server
	AT+FTPPW=?	Zobrazí nastavené heslo pro přihlášení na FTP server
AT+GPRSAPN	AT+GPRSAPN= “internet.t-mobile.cz“	Nastaví název APN pro GPRS připojení
	AT+GPRSAPN=?	Vypíše název aktuálně nastavené APN
AT+GPRSUN	AT+GPRSUN=“wap“	Nastaví uživatelské jméno pro GPRS připojení
	AT+GPRSUN=?	Vypíše nastavené uživatelské jméno GPRS připojení
AT+GPRSPW	AT+GPRSPW=“wap“	Nastaví heslo pro GPRS připojení
	AT+GPRSPW=?	Vypíše nastavené heslo GPRS připojení

4.1. Funkce použité v tomto programu

adl_simSubscribe	funkce pro přihlášení SIM karty do GSM sítě
adl_atCmdCreate	funkce umožňující zadání AT příkazu
adl_atCmdSubscribe	funkce umožňující vytvoření vlastního AT příkazů
adl_tmrSubscribe	funkce pro přihlášení časovače a jeho nastavení
adl_tmrUnSubscribe	funkce pro odhlášení časovače
adl_ioSubscribe	funkce pro ovládání GPIO portů modemu
adl_ioRead	funkce pro čtení log. hodnoty GPIO portu
adl_ioWrite	funkce pro zapsání log. hodnoty GPIO portu
adl_atSendResponse	funkce vypisující odezvu do okna Terminálu
adl_flhSubscribe	funkce pro definování objektů v paměti flash
adl_flhExist	funkce ke zjištění existence objektu v paměti flash
adl_flhRead	funkce pro zápis dat do paměti flash
adl_flhWrite	funkce ke čtení z paměti flash
ed_Init	funkce pro inicializaci knihovny pro IP spojeni
ed_GprsSetConfig	funkce k nastavení parametrů GPRS
ed_DialupConnectionStart	funkce sloužící k navázání GPRS spojení
ed_DialupConnectionStop	funkce sloužící k ukončení GPRS spojení
ed_FTPSetConfig	funkce k nastavení připojení k FTP serveru
ed_FTPPutFileSetConfig	funkce k nastavení parametrů pro přenos na FTP server
ed_FTPPut	funkce povolující přenos dat na FTP server
ed_SendData	funkce pro přenos dat

4.2. Popis vlastní aplikace a její funkcí

Při otevření programu pomocí "Open AT Project Wizard", musí být do pole "*wmnew skript additional options*" vepsán tento řádek pro začlenění eDLib knihovny do programu (viz kap. 2.3).

-otherlib /TCPIP/v3.00 -inc /TCPIP/v3.00/inc

Dále do souboru **teplomer.mak** nacházejícího se v hlavní složce programu musí být pod řádek "**EXTERNAL_LIB_LIST** = \" pro začlenění knihovny pro matematické funkce doplněn řádek:

/cygdrive/C/OpenAT/Tools/GCC/arm-elf/lib/thumb/interwork/libm.a \

4.2.1. Deklarace hlavičkových souborů

```
#include "adl_Global.h"
#include "math.h"
#include "ed_gprs.h"
#include "ed_dialup.h"
#include "ed_ftp.h"
#include "ed_msgcodes.h"
```

Soubor **adl_Global.h** odkazuje na všechny hlavičkové soubory ADL knihovny (viz. kap. 2.1.2.). Soubor **math.h** umožňuje využívat matematické funkce v programu. Hlavičkový soubor **ed_dialup.h** obsahuje funkce k uskutečnění spojení v GSM síti, **ed_gprs.h** obsahuje funkce pro nastavení a pro přenos dat prostřednictvím GPRS a **ed_ftp.h** obsahuje funkce pro nastavení a pro přenos dat pomocí protokolu FTP. A nakonec soubor **ed_msgcodes.h** slouží pro převod návratových hodnot funkcí na textové řetězce.

Dále je potřeba deklarovat proměnné nezbytné pro chod programu.

```
u32 wm_apmCustomStack [256];
const u16 wm_apmCustomStackSize=sizeof(wm_apmCustomStack);
```

4.2.2. Deklarace vlastních proměnných

```
ed_gprsSetupParams_t ParamsGPRS, DefaultParamsGPRS;
ed_FTPSetupParams_t ParamsFTP, DefaultParamsFTP;
ed_FTPPutFileParams_t ParamsFile;
adl_tmr_t *timer_mereni, *timer_ftp;
s32 un_timer_mereni, un_timer_ftp;
s8 stav555, stavled;
u32 precti555;
s8 vypis=0;
s16 pocet=0;
s16 d,f;
float teplota;
ascii asciiteplota[10];
u32 ftp_timervalue;
ascii asciicas[6]="xx:xx";
ascii asciidatum[11]="xx.xx.20xx";
ascii jmenosouboru[17]="xxxxxx_xxxx.txt";
ascii *Ftp_Handle="ftphandle", *GPRS_Handle="gprshandle",*Time_Handle="timehandle";
```

Proměnná **ParmasGPRS** a **DefaultParamsGPRS** slouží k uložení parametrů pro GPRS připojení.
Proměnná **ParamsFTP** a **DefaultParamsFTP** slouží pro uložení parametrů pro spojení s FTP serverem.

Proměnná **ParamsFile** slouží pro uložení cesty a názvu souboru na vzdáleném FTP serveru, kam budou posílána data.
Proměnné ***timer_mereni** a ***timer_ftp** ukazují na funkci **adl_tmrSubscribe**, která slouží pro přihlášení a nastavení časovače.
Proměnné **un_timer_mereni** a **un_timer_ftp** volají funkci **adl_tmrUnSubscribe**, která odhlašuje časovače **timer_mereni** a **timer_ftp**.
Proměnné **stav555** a **stavled** volají funkci **adl_ioSubscribe** sloužící k nastavení portu GPIO4 a GPIO5.
Proměnná **precti555** slouží k uložení návratové hodnoty funkce **adl_ioRead**, která načte aktuální stav portu GPIO4, ke kterému je připojen výstup časovače 555.
Proměnná **vypis** je pomocná proměnná, sloužící k tomu, aby došlo k výpočtu naměřené teploty po skončení uplynulého měření právě jedenkrát.
Proměnná **pocet** slouží ke zjištění délky pulzu z výstupu časovače 555 k určení naměřené teploty.
Proměnné **d** a **f** jsou pomocné proměnné pro převod reálného čísla do textového řetězce.
V Proměnné **teplota** je uložena aktuálně naměřená teplota.
Proměnná **ftp_timervalue** obsahuje časový interval, jak často dochází k přenosu naměřené teploty s časem měření na FTP server.
Proměnné **asciiteplota, asciicas, asciidatum** slouží k uložení naměřené teploty, času a datu měření ve formě textového řetězce.
V proměnná **jmenosouboru** je uložen název jména souboru na FTP serveru, kam se budou posílat data.
Proměnné **Ftp_Handle**, **GPRS_Handle** a **Time_Handle** obsahují názvy handlerů (obslužných rutin) pro práci s objekty v paměti flash.

4.2.3. Popis funkcí programu

```
void Gpio_TimerHandler(u8 timerId)
{
	precti555=adl_ioRead(stav555);	// precteni stavu 555
	if(precti555==0)
	{
		if(vypis==1)
		{
			teplota=(-15.939)*log(pocet)+89.343;
			d=(s16)teplota*100;
			f=(teplota*100)-d;
			if (f>24 && f<75) f=50;
			if (f<25) f=0;
			if (f>74) {f=0; teplota+=1;}
			wm_sprintf(asciiteplota, "%d.%02d", (s16)teplota , f);
			vypis=0;
			pocet=0;
		}
	}
	else
	{
		pocet++;
		vypis=1;
	}
}
```

Funkce **Gpio_TimerHandler** je volána při přetečení časovače **timer_mereni** a dochází při její provedení pomocí funkce **adl_ioRead** k přečtení stavu portu GPIO4, ke kterému

je připojen výstup časovače 555. Je-li na vstupu pinu log.úroveň H, je provedena inkrementace proměnné **pocet**, jejíž hodnota značí délku pulzu. Je-li na vstupu log. úroveň L, dojde k převodu proměnné **pocet** podle vztahu (3.4) na teplotu. Naměřená teplota je zaokrouhlována na poloviny celého čísla.

```
void adl_main(adl_InitType_e InitType)
{
        adl_simSubscribe(SimHandler, NULL");
}
```

Funkce **adl_main** je hlavní funkce a je vždy volána jako první při spuštění aplikace. Funkce **adl_simSubscribe** slouží k přihlášení SIM karty do GSM sítě.
Parametry této funkce jsou následující:
První parametr - SimHandler vrací stav inicializace SIM karty.
Druhý parametr je PIN kód, který se zadává v uvozovkách. Je-li zadávání PINu vypnuto, stačí napsat prázdné uvozovky nebo hodnotu NULL.

```
void SimHandler(u8 Event)
{
        switch(Event) // vypise, zda byla SIM inicializovana, a zda byl spravne zadany PIN kod
        {
                case ADL_SIM_EVENT_PIN_OK:
                adl_atSendResponse(ADL_AT_UNS, "\r\nPIN kod byl zadan spravne\r\n");
                break;

                case ADL_SIM_EVENT_REMOVED:
                adl_atSendResponse(ADL_AT_UNS, "\r\nNeni vlozena SIM karta\r\n");
                break;

                case ADL_SIM_EVENT_INSERTED:
                adl_atSendResponse(ADL_AT_UNS, "\r\nSIM karta je vlozena\r\n");
                break;

                case ADL_SIM_EVENT_FULL_INIT:
                adl_atSendResponse(ADL_AT_UNS, "\r\nInicializace SIM probehla v poradku\r\n");

                adl_flhSubscribe (Ftp_Handle, 3);
                adl_flhSubscribe (GPRS_Handle, 3);
                adl_flhSubscribe (Time_Handle, 1);

                adl_atCmdSubscribe("at+ftptime",AtFTPtime,ADL_CMD_TYPE_TEST|
                ADL_CMD_TYPE_PARA|0x0011);
                adl_atCmdSubscribe("at+ftpserv",AtFTPserv,ADL_CMD_TYPE_TEST|
                ADL_CMD_TYPE_PARA|0x0011);
                adl_atCmdSubscribe("at+ftpun",AtFTPun,ADL_CMD_TYPE_TEST|
                ADL_CMD_TYPE_PARA|0x0011);
                adl_atCmdSubscribe("at+ftppw",AtFTPpw,ADL_CMD_TYPE_TEST|
                ADL_CMD_TYPE_PARA|0x0011);
                adl_atCmdSubscribe("at+gprsapn",AtGPRSapn,ADL_CMD_TYPE_TEST|ADL_CMD
                _TYPE_PARA|0x0011);
                adl_atCmdSubscribe("at+gprsun",AtGPRSun,ADL_CMD_TYPE_TEST|ADL_CMD_T
                YPE_PARA|0x0011);
                adl_atCmdSubscribe("at+gprspw",AtGPRSpw,ADL_CMD_TYPE_TEST|ADL_CMD_
                TYPE_PARA|0x0011);
```

```
                ed_Init(); //inicializace knihovny pro IP spojeni

                // nastaveni portu GPIO4 jako vstup pro snimani stavu 555 a GPIO5 pro ovladani LED
                stav555=adl_ioSubscribe(ADL_IO_Q24X6_GPIO_4, 0xFFFFFFFF, 0, 0,
                (adl_ioHdlr_f) NULL);
                stavled=adl_ioSubscribe(ADL_IO_Q24X6_GPIO_5, 0, 0xFFFFFFFF, 0, (adl_ioHdlr_f)
                NULL);

                Config();

                Timerstart();

                adl_atSendResponse(ADL_AT_UNS, "\r\nMereni teploty bylo spusteno\r\n");
                break;

                case ADL_SIM_EVENT_PIN_ERROR:
                adl_atSendResponse(ADL_AT_UNS, "\r\nSpatne zadany PIN kod\r\n");
                break;

                case ADL_SIM_EVENT_PIN_NO_ATTEMPT:
                adl_atSendResponse(ADL_AT_UNS, "\r\nZbyva posledni pokus pro spravne zadani
                PIN kodu\r\n");
                break;

                case ADL_SIM_EVENT_PIN_WAIT:
                adl_atSendResponse(ADL_AT_UNS, "\r\nArgument PinCode je nastaven na
                NULL\r\n");
                break;
        }
}
```

Funkce **SimHandler** slouží jako obslužná rutina funkci **adl_simSubscribe** a podle hodnoty parametru Event vypíše, zda byla SIM karta inicializována v pořádku a zda byl správně zadán PIN.
Pokud ano, dojde k zavolání funkce **adl_flhSubscribe**, která umožní programu využití flash paměti modemu k ukládání a načítaní dat.
První parametr funkce udává jméno obslužné rutiny (handler) pro práci z pamětí flash.
Druhý parametr maximální počet objektů pro daný handler.
Funkce **adl_atCmdSubscribe** slouží k vytvoření nového AT příkazu.
První parametr udává název AT příkazu.
Druhý parametr udává název funkce, která bude volána po zadání AT příkazu.
Třetí parametr udává typ AT příkazu. V tomto případě se jedná o typ, ve kterém bude za příkazem uveden parametr z hodnotou (ADL_CMD_TYPE_PARA) a také typ sloužící ke zjištění aktuálně nastaveného parametru (ADL_CMD_TYPE_TEST). Číslo 0x00ba udává minimální (a) a maximální (b) počet parametrů v AT příkazu.
Funkce **ed_Init** slouží k inicializaci knihovny pro IP spojení.
Pro nastavení portu GPIO4 a GIPIO5 slouží funkce **adl_ioSubscribe**.
První parametr udává typ portu obsaženého v modulu a jeho číslo.
Druhý parametr udává, zda se bude zvolený port chovat jako vstup (0xFFFFFFFF), nebo výstup (0).
Třetí parametr udává jaká hodnota se má na port zapsat při spuštění - log.1 (0xFFFFFFFF) nebo log.0 (0).
Čtvrtý parametr udává časový interval mezi dvěmi vyvolávajícími operacemi s GPIO, pokud je port nastaven jako vstup. V našem případě je nastaven na 0, protože vyvolávací čas (polling time) není požadován.

Pátý parametr udává handler přijímající stav portu. Pokud je vyvolávací čas (polling time) nastaven na 0, musí být hodnota tohoto handleru NULL.
Na konec dojde k zavolání funkci **Config()** a **Timerstart()**.

```
void Timerstart(void)
{
	// nastaveni rychlosti snimani
	timer_mereni=adl_tmrSubscribe(TRUE, 1, ADL_TMR_TYPE_TICK, Gpio_TimerHandler);
	timer_ftp=adl_tmrSubscribe(FALSE, ftp_timervalue, ADL_TMR_TYPE_100MS,
	Ftp_TimerHandler);
	adl_ioWrite(stavled, ADL_IO_Q24X6_GPIO_5, 0xFFFFFFFF); //zhasne LED
}
```

Funkce **Timerstart** slouží ke spuštění časovačů pro měření teploty a pro periodické posílání naměřené teploty na FTP server. Funkce **adl_tmrSubscribe** slouží pro přihlášení a nastavení časovače.
První parametr udává, zda je časovač cyklický (TRUE), nebo ne (FALSE).
Druhý parametr udává počet period, po kterých časovač vyprší.
Třetí parametr udává typ časovače. V našem případě 100 ms nebo 18,5 ms (Tick).
Čtvrtý parametr udává, která funkce bude po vypršení časovače volána.
Funkce **adl_ioWrite** slouží pro nastavení log.1 (0xFFFFFFFF) na port GPIO5 pro zhasnutí LED diody.
První parametr udává návratovou hodnotu funkce **adl_ioSubscribe**.
Druhý parametr udává typ portu obsaženého v modulu a jeho číslo.
Třetí parametr udává jaká hodnota se má na port zapsat.

```
void Config(void)
{
	s32 length;
	ascii buffer[25];

	ParamsGPRS.Cid=1;
	ParamsGPRS.Mode=1;	//GPRS konfigurace
	wm_strcpy(DefaultParamsGPRS.ApnServ,"internet.t-mobile.cz");
	wm_strcpy(DefaultParamsGPRS.ApnUn,"wap");
	wm_strcpy(DefaultParamsGPRS.ApnPw,"wap");
	ParamsFTP.FtpPort=21;
	ParamsFTP.FtpType='A';  //ASCII soubor
	wm_strcpy(DefaultParamsFTP.FtpServ,"88.86.103.242");
	wm_strcpy(DefaultParamsFTP.FtpUn,"openat-teplota");
	wm_strcpy(DefaultParamsFTP.FtpPw,"qwert");

	if ((length = adl_flhExist(Ftp_Handle, 0))>0)
		adl_flhRead(Ftp_Handle,0, length, ParamsFTP.FtpServ);
	else
	{
		wm_strcpy(ParamsFTP.FtpServ,DefaultParamsFTP.FtpServ);
		adl_flhWrite (Ftp_Handle, 0, strlen(DefaultParamsFTP.FtpServ),
		DefaultParamsFTP.FtpServ );
	}

	if ((length = adl_flhExist(Ftp_Handle, 1))>0)
		adl_flhRead(Ftp_Handle,1, length, ParamsFTP.FtpUn);
	else
```

```
        {
                wm_strcpy(ParamsFTP.FtpUn,DefaultParamsFTP.FtpUn);
                adl_flhWrite (Ftp_Handle, 1, strlen(DefaultParamsFTP.FtpUn),
                DefaultParamsFTP.FtpUn );
        }

        if ((length = adl_flhExist(Ftp_Handle, 2))>0)
                adl_flhRead(Ftp_Handle,2, length, ParamsFTP.FtpPw);
        else
        {
                wm_strcpy(ParamsFTP.FtpPw,DefaultParamsFTP.FtpPw);
                adl_flhWrite (Ftp_Handle, 2, strlen(DefaultParamsFTP.FtpPw),
                DefaultParamsFTP.FtpPw );
        }

        if ((length = adl_flhExist(GPRS_Handle, 0))>0)
                adl_flhRead(GPRS_Handle,0, length, ParamsGPRS.ApnServ);
        else
        {
                wm_strcpy(ParamsGPRS.ApnServ,DefaultParamsGPRS.ApnServ);
                adl_flhWrite (GPRS_Handle, 0, strlen(DefaultParamsGPRS.ApnServ),
                DefaultParamsGPRS.ApnServ );
        }

        if ((length = adl_flhExist(GPRS_Handle, 1))>0)
                adl_flhRead(GPRS_Handle,1, length, ParamsGPRS.ApnUn);
        else
        {
                wm_strcpy(ParamsGPRS.ApnUn,DefaultParamsGPRS.ApnUn);
                adl_flhWrite (GPRS_Handle, 1, strlen(DefaultParamsGPRS.ApnUn),
                DefaultParamsGPRS.ApnUn);
        }

        if ((length = adl_flhExist(GPRS_Handle, 2))>0)
                adl_flhRead(GPRS_Handle,2, length, ParamsGPRS.ApnPw);
        else
        {
                wm_strcpy(ParamsGPRS.ApnPw,DefaultParamsGPRS.ApnPw);
                adl_flhWrite (GPRS_Handle, 2, strlen(DefaultParamsGPRS.ApnPw),
                DefaultParamsGPRS.ApnPw);
        }

        if ((length = adl_flhExist(Time_Handle, 0))>0)
        {
                adl_flhRead(Time_Handle,0, length, buffer);
                ftp_timervalue=600*wm_atoi(buffer);
        }
        else
        {
                ftp_timervalue=600;
        }
}
```

Funkce **Config** slouží k zadání GPRS a FTP parametrů.
Proměnně **ParamsGPRS** a **DefaultParamsGPRS** jsou struktury typu ed_gprsSetupParams_t a obsahují parametry pro připojení GPRS. Parametr **ParamsGPRS.Cid=1** definuje PDP kontext. Parametr **ParamsGPRS.Mode=1** značí, že je pro přenos dat použit mód GPRS (0 by znamenala GSM mód).

Proměnně **ParamsFTP** a **DefaultParamsFTP** jsou struktury typu ed_FTPSetupParams_t a obsahují parametry pro konfigurace FTP spojení. Parametr **ParamsFTP.FtpPort=21** určuje, že se ke vzdálenému FTP serveru aplikace bude připojovat přes port 21. Parametr **ParamsFTP.FtpType='A'** znamená, že přenos dat bude probíhat v režimu ASCII. Parametr **ParamsFTP.FtpServ** obsahuje IP adresu nebo doménovou adresu FTP serveru. V parametru **ParamsFTP.FtpUn** je uložené uživatelské jméno a v parametru **ParamsFTP.FtpPw** uživatelské heslo pro přihlášení k FTP serveru.
Funkce **adl_flhExist** zkoumá existenci objektu v paměti flash. Pokud objekt existuje vrací jeho délku. Jinak vrací 0.
Její první parametr udává jméno obslužné rutiny (handleru).
Druhý parametr udává číslo objektu zvoleného handleru.
Funkce **adl_flhRead** slouží ke čtení z paměti flash.
První parametr udává jméno handleru.
Druhý parametr udává číslo objektu pro zvolený handler.
Třetí parametr udává délku objektu.
Čtvrtý parametr název proměnné, do které budou data z paměti uložena.
Funkce **adl_flhWrite** slouží pro zápis do flash paměti .
První parametr udává jméno handleru.
Druhý parametr udává číslo objektu pro daný handler.
Třetí parametr udává délku objektu.
Čtvrtý parametr název proměnné, z které budou údaje do paměti uložena.
Proměnné **DefaultParamsFTP** a **DefaultParamsGPRS** slouží pro nastavení konfiguračních parametrů, nejsou-li nalezeny potřebné objekty v paměti flash.
Proměnná **ftp_timervalue** obsahuje časový interval, jak často bude docházet k přenosu naměřené teploty s časem měření na FTP server.

```
void AtFTPtime(adl_atCmdPreParser_t *paras)
{
        ascii buffer[25];
        un_timer_ftp=adl_tmrUnSubscribe ( timer_ftp, Ftp_TimerHandler, ADL_TMR_TYPE_100MS
        );
        un_timer_mereni=adl_tmrUnSubscribe ( timer_mereni, Gpio_TimerHandler,
        ADL_TMR_TYPE_TICK );
        switch(paras->Type)
        {
                case ADL_CMD_TYPE_TEST:
                wm_sprintf(buffer, "%d", (ftp_timervalue/600));
                break;

                case ADL_CMD_TYPE_PARA:
                wm_strGetParameterString ( buffer, paras->StrData, 1 );
                ftp_timervalue=600*wm_atoi(buffer); //prevod minut na sekundy
                adl_flhWrite (Time_Handle, 0, strlen(buffer), buffer );
                break;
        }
        adl_atSendResponse(ADL_AT_RSP, "\r\n");
        adl_atSendResponse(ADL_AT_RSP, "Namerena teplota je posilana na FTP server s periodou ");
        adl_atSendResponse(ADL_AT_RSP, buffer);
        adl_atSendResponse(ADL_AT_RSP, " minut\r\n");
        Timerstart();
}

void AtFTPserv(adl_atCmdPreParser_t *paras)
```

```
{
        ascii buffer[25];
        switch(paras->Type)
        {
                case ADL_CMD_TYPE_PARA:
                wm_strGetParameterString ( buffer, paras->StrData, 1 );
                wm_strcpy(ParamsFTP.FtpServ,buffer);
                adl_flhWrite (Ftp_Handle, 0, strlen(buffer), buffer );

                break;
        }

        adl_atSendResponse(ADL_AT_RSP, "\r\nFTP server: ");
        adl_atSendResponse(ADL_AT_RSP, ParamsFTP.FtpServ);
        adl_atSendResponse(ADL_AT_RSP, "\r\n");
}

void AtFTPun(adl_atCmdPreParser_t *paras)
{
        ascii buffer[25];
        switch(paras->Type)
        {
                case ADL_CMD_TYPE_PARA:
                wm_strGetParameterString ( buffer, paras->StrData, 1 );
                wm_strcpy(ParamsFTP.FtpUn,buffer);
                adl_flhWrite (Ftp_Handle, 1, strlen(buffer), buffer );
                break;
        }
        adl_atSendResponse(ADL_AT_RSP, "\r\nFTP uzivatel: ");
        adl_atSendResponse(ADL_AT_RSP, ParamsFTP.FtpUn);
        adl_atSendResponse(ADL_AT_RSP, "\r\n");
}


void AtFTPpw(adl_atCmdPreParser_t *paras)
{
        ascii buffer[25];
        switch(paras->Type)
        {
                case ADL_CMD_TYPE_PARA:
                wm_strGetParameterString ( buffer, paras->StrData, 1 );
                wm_strcpy(ParamsFTP.FtpPw,buffer);
                adl_flhWrite (Ftp_Handle, 2, strlen(buffer), buffer );
                break;
        }
        adl_atSendResponse(ADL_AT_RSP, "\r\nFTP heslo: ");
        adl_atSendResponse(ADL_AT_RSP, ParamsFTP.FtpPw);
        adl_atSendResponse(ADL_AT_RSP, "\r\n");
}

void AtGPRSapn(adl_atCmdPreParser_t *paras)
{
        ascii buffer[25];
        switch(paras->Type)
        {
                case ADL_CMD_TYPE_PARA:
                wm_strGetParameterString ( buffer, paras->StrData, 1 );
                wm_strcpy(ParamsGPRS.ApnServ,buffer);
                adl_flhWrite (GPRS_Handle, 0, strlen(buffer), buffer );
                break;
```

```
        }
        adl_atSendResponse(ADL_AT_RSP, "\r\nAdresa APN: ");
        adl_atSendResponse(ADL_AT_RSP, ParamsGPRS.ApnServ);
        adl_atSendResponse(ADL_AT_RSP, "\r\n");
}

void AtGPRSun(adl_atCmdPreParser_t *paras)
{
        ascii buffer[25];
        switch(paras->Type)
        {
                case ADL_CMD_TYPE_PARA:
                wm_strGetParameterString ( buffer, paras->StrData, 1 );
                wm_strcpy(ParamsGPRS.ApnUn,buffer);
                adl_flhWrite (GPRS_Handle, 1, strlen(buffer), buffer );
                break;
        }
        adl_atSendResponse(ADL_AT_RSP, "\r\nAPN uzivatel: ");
        adl_atSendResponse(ADL_AT_RSP, ParamsGPRS.ApnUn);
        adl_atSendResponse(ADL_AT_RSP, "\r\n");
}

void AtGPRSpw(adl_atCmdPreParser_t *paras)
{
        ascii buffer[25];
        switch(paras->Type)
        {
                case ADL_CMD_TYPE_PARA:
                wm_strGetParameterString ( buffer, paras->StrData, 1 );
                wm_strcpy(ParamsGPRS.ApnPw,buffer);
                adl_flhWrite (GPRS_Handle, 2, strlen(buffer), buffer );
                break;
        }
        adl_atSendResponse(ADL_AT_RSP, "\r\nAPN heslo: ");
        adl_atSendResponse(ADL_AT_RSP, ParamsGPRS.ApnPw);
        adl_atSendResponse(ADL_AT_RSP, "\r\n");
}
```

Funkce **AtFTPtime**, **AtFTPserv**, **AtFTPun**, **AtFTPpw, AtGPRSapn, AtGPRSun** a **AtGPRSpw** jsou popsány společně, protože jsou si podobné a využívají stejné funkce.
Tyto funkce jsou volány po zadání AT příkazů **AT+FTPTIME**, **AT+FTPSERV**, **AT+FTPUN**, **AT+FTPPW, AT+GPRSAPN, AT+GPRSUN** a **AT+GPRSPW** v okně HyperTerminalu.
Funkce **adl_tmrUnSubscribe** slouží k odhlášení časovače.
První parametr udává název časovače, který bude odhlášen.
Druhý parametr udává funkci, která byla vyvolávána po přetečení časovače.
Třetí parametr udává typ časovače.
Je-li AT příkaz zadán jako testovací, tedy bez parametrů (např. AT+FTPTIME=?), dojde pouze k výpisu aktuálně nastavených parametrů.
Je-li AT příkaz doplněn parametrem (např. AT+FTPTIME=20), dojde ke změně hodnoty zvolené proměnné a také zápisu nově nastaveného parametru do flash paměti.
Funkce **wm_strGetParameterString** slouží k získání parametrů z AT příkazů jako textové řetězce. Více parametrů jsou oddělené čárkami a indexují se od 1 do n, kde n je celkový počet parametrů.

```
void Ftp_TimerHandler(u8 timerId)
{
	un_timer_mereni=adl_tmrUnSubscribe ( timer_mereni, Gpio_TimerHandler,
	ADL_TMR_TYPE_TICK );
	adl_atCmdCreate("AT+CCLK?",TRUE,ClockHandler,"+CCLK",NULL);
	adl_tmrSubscribe(FALSE,10,ADL_TMR_TYPE_100MS, Ftp_Config_TimerHandler);
}
```

Funkce **Ftp_TimerHandler** je volána vždy po vypršení časovače **timer_ftp**. Během ní dojde k odhlášení časovače pro měření teploty **timer_mereni** pomocí funkce **adl_tmrUnSubscribe**. Program zjistí aktuální čas a datum zadáním AT příkazu AT+CCLK? pomocí funkce **adl_atCmdCreate** a odpověď bude předána jako vstupní parametr funkci **ClockHandler**.
První parametr udává volaný AT příkaz.
Druhý parametr udává zda se má odpověď vypsat do okna terminálu.
Třetí parametr udává funkci, které bude předána odpověď AT příkazu jako její vstupní parametr.
Časovač je zde použit z toho důvodu, protože provedení AT příkazu trvá určitou dobu, a proto je zde nutné provést časové zdržení programu.

```
void Ftp_Config_TimerHandler (u8 timerId)
{
	s8 sreturnCode = ERROR;

	wm_strcpy(ParamsFile.FtpPutFilename,jmenosouboru);
	wm_strcpy(ParamsFile.FtpPutPath,"/mereni/");

	adl_atCmdCreate("AT+CGATT=1",TRUE,(adl_atRspHandler_t)NULL,NULL);
	adl_ioWrite(stavled, ADL_IO_Q24X6_GPIO_5, 0); //rozviti se LED

	sreturnCode=ed_GprsSetConfig(&ParamsGPRS);
	if (sreturnCode == 0)
	{
		//Parametry OK
		adl_atSendResponse(ADL_AT_UNS, "GPRS parametry OK\n\r");
		adl_tmrSubscribe(FALSE,50,ADL_TMR_TYPE_100MS, Send_TimerHandler);
	}
	else
	{
		adl_atSendResponse(ADL_AT_UNS, "GPRS parametry ERROR\n\r");
		Timerstart();
	}
}
```

Po vypršení časovače dojde k zavolání funkce **Ftp_Config_TimerHandler**. Parametr **ParamsFile.FtpPutFilename** obsahuje název souboru, ve kterém budou posílána data a parametr **ParamsFile.FtpPutPath** cestu na FTP serveru, kde se soubor nachází. Po zadaní příkazu AT+CGATT=1 modemu oznámíme, že se chystá GPRS připojení a pomocí funkce **adl_ioWrite** dojde k rozsvícení LED diody. Pomocí funkce **ed_GprsSetConfig** se provede nastavení parametrů pro GPRS a do proměnné **sreturnCode** se uloží návratová hodnota této funkce. Pokud bude proměnná **sreturnCode** rovná 0, jsou parametry zadány správně a dojde ke spuštění časovače pro časové zpoždění. a po jeho vypršení se zavolá funkce **Send_TimerHandler**. Pokud

bude v proměnné **sreturnCode** jiná hodnota, nedojde ke spojení a obnoví se proces měření.

```
void Send_TimerHandler(u8 timerId)
{
	s8 sreturnCode=ERROR;
	sreturnCode=ed_DialupConnectionStart(DialupHandler);
	if(sreturnCode==0)
	{
		adl_atSendResponse(ADL_AT_UNS, "Diaul up OK\n\r");
	}
	else
	{
		adl_atSendResponse(ADL_AT_UNS, "Diaul up ERROR\n\r");
		Timerstart();
	}
}
```

V těle funkce **Send_TimerHandler** je volána funkce **ed_DialupConnectionStart** sloužící k navázání GPRS spojení. Po pokusu o spojení je volána funkce **DialupHandler**. Pokud bude proměnná **sreturnCode** rovná 0, bylo spojení navázáno v pořádku. Pokud bude v proměnné jiná hodnota, nedošlo ke spojení a obnoví se proces měření.

```
void DialupHandler( s32 ResponseCode )
{
	switch (ResponseCode)
	{
		case ED_OK_GPRS_SESSION_SET: // GPRS online
		adl_atSendResponse(ADL_AT_UNS, "GPRS pripojeno\n\r");
		ConectFTP();
		break;
		default:
		//Errorr GPRS
		adl_atSendResponse(ADL_AT_UNS, "Chyba pri GPRS pripojeni\n\r");
		Timerstart();
		break;
	}
}
```

Po pokusu o GPRS spojení pomocí funkce **ed_DialupConnectionStart** dojde k volání funkce **DialupHandler.** Jejím jediným parametrem je proměnná **ResponseCode**, ve které je uloženo, zda bylo spojení úspěšné, či nikoliv. Pokud bylo spojení provedeno úspěšně, zavolá se funkce **ConectFTP**. Pokud bylo neúspěšné, obnoví se proces měření.

```
void ConectFTP(void)
{
	s8 sreturnCode=ERROR;
	sreturnCode=ed_FTPSetConfig(&ParamsFTP);
	if(sreturnCode==0)
	{
		//FTP konfigurace ok
		adl_atSendResponse(ADL_AT_UNS, "FTP konfigurace uspesna\n\r");
		sreturnCode=ed_FTPPutFileSetConfig(&ParamsFile);
```

```
                if(sreturnCode==0)
                {
                        adl_atSendResponse(ADL_AT_UNS, "FTP_PUT konigurace uspesna\n\r");
                        ed_FTPPut(FtpPutHandler,FtpPutData);
                }
                else
                {
                        adl_atSendResponse(ADL_AT_UNS, "FTP_PUT konigurace neuspesna\n\r");
                        Timerstart();
                }
        }
        else
        {
                //FTP konfigurace ERRORR
                adl_atSendResponse(ADL_AT_UNS, "FTP konfigurace neuspesna\n\r");
                Timerstart();
        }
}
```

Při běhu funkce **ConectFTP** dochází k volání funkce **ed_FTPSetConfig**, která provede nastavení parametrů pro FTP připojení. Do proměnné **sreturnCode** se uloží návratová hodnota této funkce. Pokud bude proměnná **sreturnCode** rovná 0, jsou parametry zadány správně a dojde k volání funkce **ed_FTPPutFileSetConfig**, která nastaví parametry FTP spojení tak, aby bylo možné přenést data na FTP server. Pokud budou parametry v pořádku, zavolá se funkce **ed_FTPPut,** která zajistí přenos dat na FTP server.
První parametr je handler této funkce **FtpPutHandler**, který slouží ke zjištění stavu přenosu dat.
Druhým parametrem funkce je volání funkce **FtpPutData** pro odeslání dat na FTP server.

```
void FtpPutHandler(s32 ResponseCode,TeDHandle id)
{
        switch (ResponseCode)
        {
        case ED_OK_FILE_TRANSFERED:
        adl_atSendResponse(ADL_AT_UNS, "Data na FTP server uspesne prenesena\n\r");
        break;
        case ED_INFO_WAITING_FOR_DATA:
        adl_atSendResponse(ADL_AT_UNS, "FTP Server ceka na prenos dat\n\r");
        break;
        default:
        adl_atSendResponse(ADL_AT_UNS, "Spatne nastavene parametry pro pripojeni k FTP
        serveru\n\r");
        ed_DialupConnectionStop (DialupHandler2);
        Timerstart();
        break;
        }

}
```

Funkce **FtpPutHandler** slouží jako obslužná rutina (handler) pro funkci **ed_FTPPut**. Kontroluje stav přenosu dat na FTP server. Nemůže-li dojít z nějakého důvodu

k přenosu dat, například z důvodu špatně nastavených parametrů FTP připojení, zavolá se funkce **ed_DialupConnectionStop,** sloužící k rozpojení GPRS spojení.

```
void FtpPutData(u16 MaxLen,TeDHandle id)
{
	u8 SmsText[60]="";
	strcat(SmsText, asciiteplota);
	strcat(SmsText, "\n");
	strcat(SmsText, asciidatum);
	strcat(SmsText, "\n");
	strcat(SmsText, asciicas);
	strcat(SmsText, "\n");

	adl_atSendResponse ( ADL_AT_RSP, "\r\nNamerena teplota je: ");
	adl_atSendResponse ( ADL_AT_RSP, asciiteplota );
	adl_atSendResponse ( ADL_AT_RSP, "\r\n" );

	adl_atSendResponse(ADL_AT_UNS, "Data jsou prenesena na FTP server\n\r");
	ed_SendData(SmsText,strlen(SmsText),TRUE);
	adl_tmrSubscribe(FALSE,50,ADL_TMR_TYPE_100MS, Send_TimerHandler2);
}
```

V těle funkce **FtpPutData** dojde k sestavení obsahu souboru, který bude poslán na FTP server. Text zprávy je uložen do proměnné **SmsText** a je složen z naměřené teploty, data a času měření. Pomocí funkce **ed_SendData** dojde k přenosu dat**.**
První parametr funkce určuje data, které chceme poslat
Druhý parametr udává délku dat. Pomocí funkce strlen(SmsText) se délka přizpůsobí délce posílaného řetězc.
Třetí parametr udává, že po odeslání dat se spojení s FTP serverem ukončí.
Časovač je zde použit z toho důvodu, aby nedošlo k odpojení GPRS spojení dříve, než všechna data budou odeslána na FTP server.

```
void Send_TimerHandler2(u8 timerId)
{
	ed_DialupConnectionStop (DialupHandler2);
	Timerstart();
}
```

Po vypršení časovače dojde ke zpuštění funkce **Send_TimerHandler2**. V ní je volána funkce **ed_DialupConnectionStop,** sloužící k rozpojení GPRS spojení. Po pokusu o GPRS rozpojení dojde k volání funkce **DialupHandler2** s parametrem **ResponseCode**, ve kterém je uloženo, zda bylo rozpojení úspěšné, či nikoliv. Nakonec dojde k volání funkce **Timerstart**, která zajistí opětovné spuštění časovače pro měření teploty.

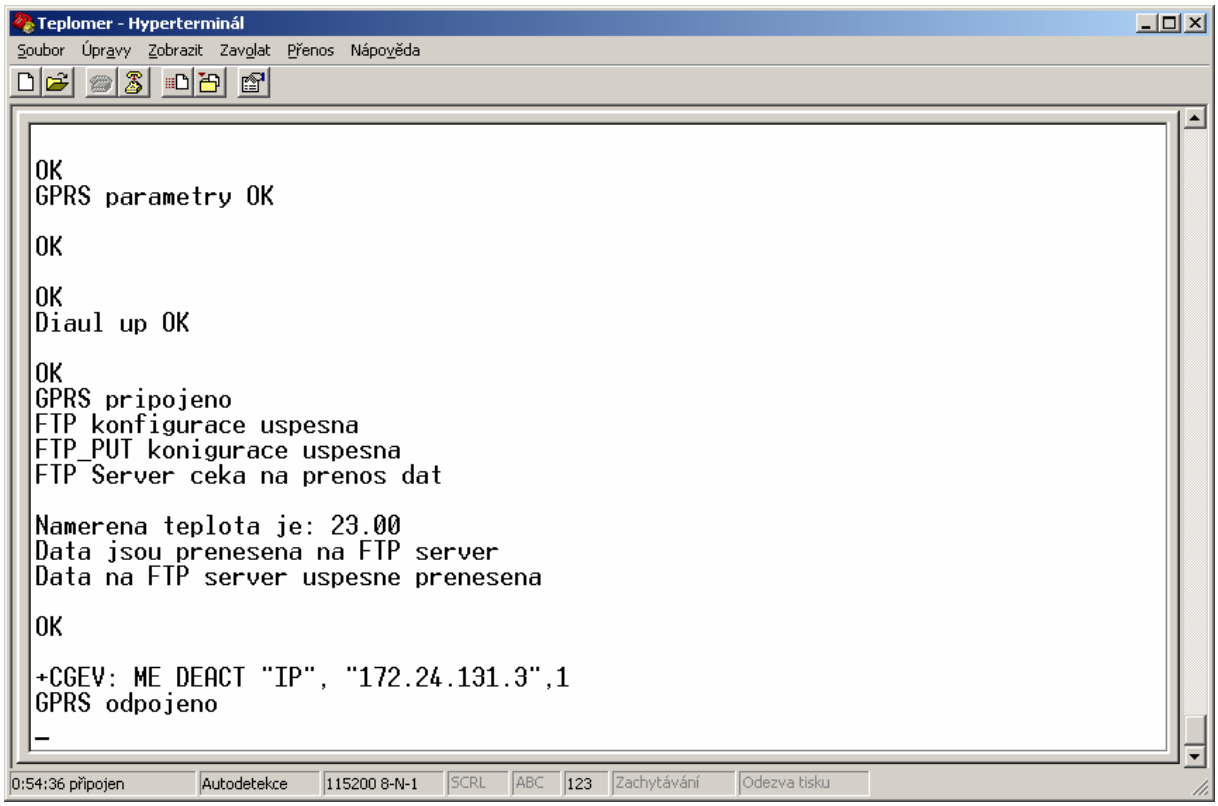
```
void DialupHandler2(s32 ResponseCode)
{
	switch (ResponseCode)
	{
		case ED_OK_ON_HOOK: // GPRS online
		adl_atSendResponse(ADL_AT_UNS, "GPRS odpojeno\n\r");
		break;
```

```
                default:
                //Errorr GPRS
                adl_atSendResponse(ADL_AT_UNS, "Chyba pri odpojeni GPRS\n\r");
                break;
        }
}
```

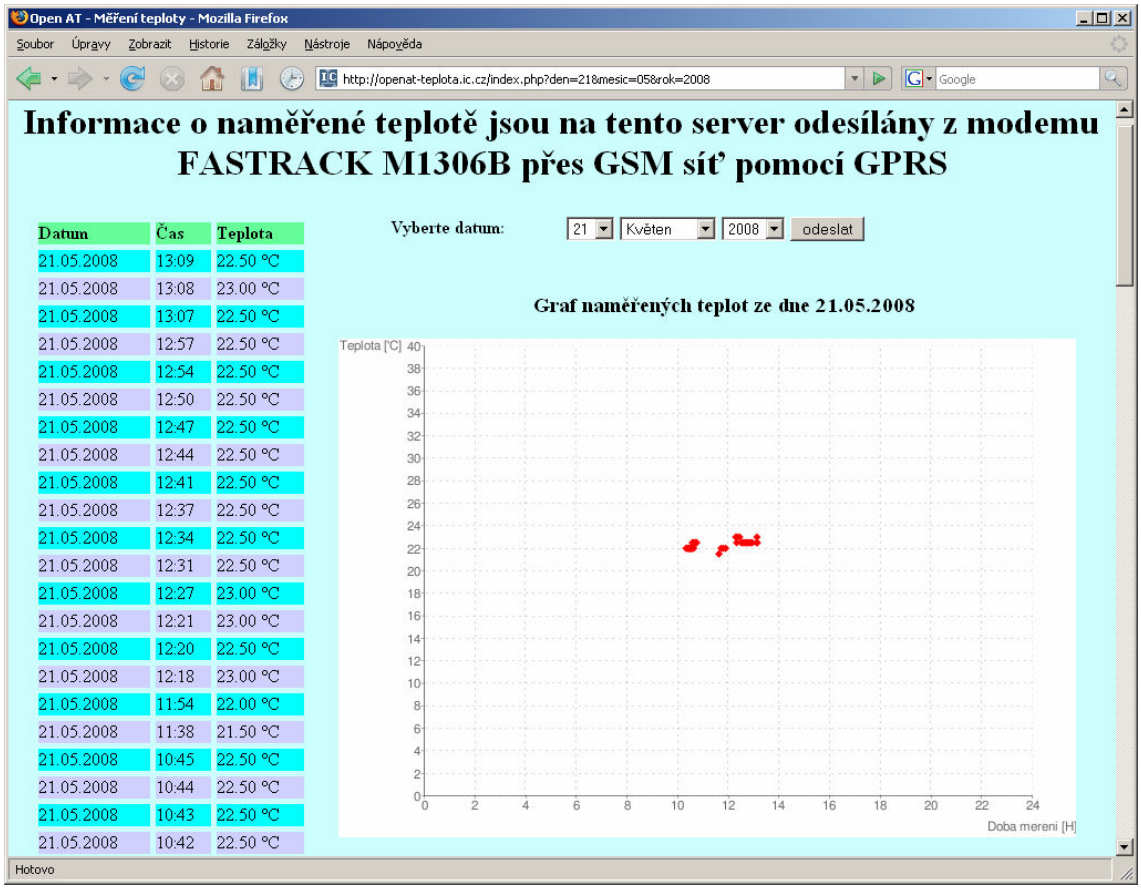
Funkce **DialupHandler2** slouží ke zjištění správného odpojení GPRS služeb**.**

```
void ClockHandler (adl_atResponse_t *resp)
{
        ascii buffer[100];
        wm_strcpy(buffer, resp->StrData );

        asciidatum[0]=buffer[16];
        asciidatum[1]=buffer[17];
        asciidatum[3]=buffer[13];
        asciidatum[4]=buffer[14];
        asciidatum[8]=buffer[10];
        asciidatum[9]=buffer[11];

        asciicas[0]=buffer[19];
        asciicas[1]=buffer[20];
        asciicas[3]=buffer[22];
        asciicas[4]=buffer[23];

        jmenosouboru[0]=buffer[10];
        jmenosouboru[1]=buffer[11];
        jmenosouboru[2]=buffer[13];
        jmenosouboru[3]=buffer[14];
        jmenosouboru[4]=buffer[16];
        jmenosouboru[5]=buffer[17];
        jmenosouboru[7]=buffer[19];
        jmenosouboru[8]=buffer[20];
        jmenosouboru[9]=buffer[22];
        jmenosouboru[10]=buffer[23];
}
```

Funkce **ClockHandler** slouží k sestavení data měření do formátu “dd.mm.rr“ a uložení do proměnné **asciidatum**. Čas měření je ve formátu “hh:mm“ a je uložen do proměnné **asciicas**. Jméno souboru posílaného na FTP server, do kterého budou uložena data s naměřenou teplotou společně s časem a datem měření, je ve formátu “rrmmdd_hhmm.txt“ a je uloženo do proměnné **jmenosouboru**.



Obr. 4.1: Ukázka aplikace pro měření teploty s přenosem na FTP server



Datum	Čas	Teplota
21.05.2008	13:09	22.50 °C
21.05.2008	13:08	23.00 °C
21.05.2008	13:07	22.50 °C
21.05.2008	12:57	22.50 °C
21.05.2008	12:54	22.50 °C
21.05.2008	12:50	22.50 °C
21.05.2008	12:47	22.50 °C
21.05.2008	12:44	22.50 °C
21.05.2008	12:41	22.50 °C
21.05.2008	12:37	22.50 °C
21.05.2008	12:34	22.50 °C
21.05.2008	12:31	22.50 °C
21.05.2008	12:27	23.00 °C
21.05.2008	12:21	23.00 °C
21.05.2008	12:20	22.50 °C
21.05.2008	12:18	23.00 °C
21.05.2008	11:54	22.00 °C
21.05.2008	11:38	21.50 °C
21.05.2008	10:45	22.50 °C
21.05.2008	10:44	22.50 °C
21.05.2008	10:43	22.50 °C
21.05.2008	10:42	22.50 °C

Obr. 4.2: Ukázka webové stránky s údaji o naměřené teplotě

5. Aplikace pro měření teploty s posíláním SMS zpráv

Způsob provedení měření teploty je stejný jako u předchozího programu. Aplikace je určena k poslání SMS zpráv v síti GSM. Obdrží-li modem SMS zprávu s textem "**openat**", odpoví na telefonní číslo, ze kterého SMS přišla zprávou s údaji o naměřené teplotě. Pro správnou funkci programu je nutné nastavit zkratovací propojku umístěnou na modulu do pozice **Tl1**. Tlačítko je použito pro ověření správné funkčnosti SMS služby na modemu tak, že po stisku dojde k odeslání SMS zprávy na číslo uvedené ve zdrojovém kódu programu. Pro správnou funkci programu, musí být zkompilovaná aplikace nahrána a spuštěna na modemu v cílovém módu (viz. kap. 2.4.2).

5.1. Funkce použité v tomto programu

adl_simSubscribe	funkce pro přihlášení SIM karty do GSM sítě
adl_tmrSubscribe	funkce pro přihlášení časovače a jeho nastavení
adl_tmrUnSubscribe	funkce pro odhlášení časovače
adl_ioSubscribe	funkce pro ovládání GPIO portů modemu
adl_ioRead	funkce pro čtení log. hodnoty GPIO portu
adl_atSendResponse	funkce vypisující odezvu do okna Terminálu
adl_smsSubscribe	funkce pro přihlášení k SMS službám
adl_smsSend	funkce pro odeslání SMS zprávy

5.2. Popis vlastní aplikace a její funkcí

Po otevření programu pomocí "Open AT Project Wizard", musí být do souboru **SMS.mak** nacházejícího se v hlavní složce programu vložen pod řádek "**EXTERNAL_LIB_LIST** = \" pro začlenění knihovny pro matematické funkce do programu tento text:

**/cygdrive/C/OpenAT/Tools/GCC/arm-elf/lib/thumb/interwork/libm.a **

5.2.1. Deklarace hlavičkových souborů

```
#include "adl_Global.h"
#include "math.h"
```

Soubor **adl_Global.h** odkazuje na všechny hlavičkové soubory ADL knihoven (viz. kap. 2.1.2.), takže je není potřeba vypisovat jednotlivě. Soubor **math.h** umožňuje využívat matematické funkce v programu.

Dále je potřeba deklarovat proměnné nezbytné pro chod programu.

```
u32 wm_apmCustomStack [256];
const u16 wm_apmCustomStackSize=sizeof(wm_apmCustomStack);
```

5.2.2. Deklarace vlastních proměnných

```
adl_tmr_t *timer_tla, *timer_mereni;
s32 un_timer_mereni, un_timer_tla;
s8 stavtla, stav555, sms, vysledek_sms, vypis=0;
s16 pocet=0, d, f;
float teplota;
u32 prectitla, precti555;
ascii asciipocet[10];
ascii asciiteplota[10];
ascii SmsText[160]="";
```

Proměnné ***timer_mereni** a ***timer_tla** ukazují na funkci **adl_tmrSubscribe**, která slouží pro přihlášení a nastavení časovače.
Proměnné **un_timer_mereni** a **un_timer_tla** volají funkci **adl_tmrUnSubscribe**, která odhlašuje časovače **timer_mereni** a **timer_tla**.
Proměnné **stav555** a **stavtla** volají funkci **adl_ioSubscribe** sloužící k nastavení portu GPIO4 a GPIO5.
Proměnná **sms** volá funkci **adl_smsSubscribe** pro přihlášení k SMS službám.
Proměnná **vysledek_sms** slouží ke kontrole, zda byla SMS odeslána v pořádku.
Proměnná **vypis** je pomocná proměnná, sloužící k tomu, aby došlo k výpisu naměřené teploty do okna terminálu po skončení jednoho měření právě jedenkrát.
Proměnná **pocet** slouží ke zjištění délky pulzu z výstupu časovače 555 k určení naměřené teploty.
Proměnné **d** a **f** jsou pomocné proměnné pro převod reálného čísla do textového řetězce.

V Proměnné **teplota** je uložena aktuálně naměřená teplota.
Proměnná **prectitla** slouží k uložení návratové hodnoty funkce **adl_ioRead**, která načte aktuální stav portu GPIO5, ke kterému je připojeno tlačítko.
Proměnná **precti555** slouží k uložení návratové hodnoty funkce **adl_ioRead**, která načte aktuální stav portu GPIO4, ke kterému je připojen výstup časovače 555.
Proměnné **asciipocet, asciiteplota** slouží k uložení délky pulzu z výstupu časovače 555 a naměřené teploty ve formě textového řetězce.
Proměnná **SmsText** je textový řetězec o délce 160 bytů, do kterého je uložen text SMS zprávy.

5.2.3. Popis funkcí programu

```
void Gpio_555_TimerHandler(u8 timerId)
{
        precti555=adl_ioRead(stav555);    // precteni stavu 555
        if (precti555==0)
        {
                if (vypis==1)
                {
                        teplota=(-15.939)*log(pocet)+89.343;
                        d=(s16)teplota*100;
                        f=(teplota*100)-d;
                        if (f>24 && f<75) f=50;
                        if (f<25) f=0;
                        if (f>74) {f=0; teplota+=1;}
                        wm_sprintf(asciiteplota, "%d.%02d", (s16)teplota , f);
                        wm_sprintf(asciipocet, "%d", pocet);
                        adl_atSendResponse(ADL_AT_UNS, "Pocet zachycenych vzorku: ");
                        adl_atSendResponse(ADL_AT_UNS, asciipocet);
                        adl_atSendResponse(ADL_AT_UNS, "\r\n");
                        adl_atSendResponse(ADL_AT_UNS, "Teplota je: ");
                        adl_atSendResponse(ADL_AT_UNS, asciiteplota);
                        adl_atSendResponse(ADL_AT_UNS, "\r\n");
                        adl_atSendResponse(ADL_AT_UNS, "\r\n");
                        vypis=0;
                        pocet=0;
                }
        }
        else
        {
                pocet++;
                vypis=1;
        }
}
```

Funkce **Gpio_555_TimerHandler** je volána při přetečení časovače **timer_mereni** a dochází při její provedení k přečtení stavu portu GPIO4, ke kterému je připojen výstup časovače 555. Je-li na vstupu pinu log.úroveň H je provedena inkrementace proměnné **pocet**, jejíž hodnota značí délku pulzu. Je-li na vstupu log. úroveň L dojde k převodu proměnné **pocet** podle vztahu (3.4) na teplotu, která je pomocí funkce **adl_atSendResponse** vypsána do okna terminálu.

```
void adl_main(adl_InitType_e InitType)
{
        // zadani PIN kodu
        adl_simSubscribe(SimHandler, NULL);
}
```

Funkce **adl_main** je hlavní funkce a je vždy volána jako první při spuštění aplikace. Funkce **adl_simSubscribe** slouží k přihlášení SIM karty do GSM sítě. Parametry této funkce jsou uvedeny u předchozího programu (kap. 4.2.3).

```
void SimHandler(u8 Event)
{
        switch (Event) // vypise, zda byla SIM inicializovana, a zda byl spravne zadany PIN kod
        {
                case ADL_SIM_EVENT_PIN_OK:
                adl_atSendResponse(ADL_AT_UNS,"\r\nPIN kod byl zadan spravne\r\n");
                break;

                case ADL_SIM_EVENT_REMOVED:
                adl_atSendResponse(ADL_AT_UNS,"\r\nNeni vlozena SIM karta\r\n");
                break;

                case ADL_SIM_EVENT_INSERTED:
                adl_atSendResponse(ADL_AT_UNS,"\r\nSIM karta je vlozena\r\n");
                break;

                case ADL_SIM_EVENT_FULL_INIT:
                adl_atSendResponse(ADL_AT_UNS,"\r\nInicializace SIM probehla v poradku\r\n");

                sms=adl_smsSubscribe(SmsHandler, SmsCtrlHandler, ADL_SMS_MODE_TEXT);

                // nastaveni portu GPIO5 ke kteremu je pripojene tlacitko
                stavtla=adl_ioSubscribe (ADL_IO_Q24X6_GPIO_5, 0xFFFFFFFF, 0, 0, (adl_ioHdlr_f)
                NULL);
                // nastaveni portu GPIO4 jako vstup pro snimani stavu 555
                stav555=adl_ioSubscribe (ADL_IO_Q24X6_GPIO_4, 0xFFFFFFFF, 0, 0,
                (adl_ioHdlr_f) NULL);

                Timerstart();
                break;

                case ADL_SIM_EVENT_PIN_ERROR:
                adl_atSendResponse(ADL_AT_UNS,"\r\nSpatne zadany PIN kod\r\n");
                break;

                case ADL_SIM_EVENT_PIN_NO_ATTEMPT:
                adl_atSendResponse(ADL_AT_UNS,"\r\nZbyva posledni pokus pro spravne zadani
                PIN kodu\r\n");
                break;

                case ADL_SIM_EVENT_PIN_WAIT:
                adl_atSendResponse(ADL_AT_UNS,"\r\nArgument PinCode je nastaven na
                NULL\r\n");
                break;
        }
}
```

Funkce **SimHandler** je handler funkce **adl_simSubscribe** a podle hodnoty parametru Event vypíše, zda byla SIM karta inicializována v pořádku, a zda byl správně zadán PIN.
Pokud inicializace SIM karty proběhla v pořádku, provede se nastavení portu GPIO4, ke kterému je připojen výstup časovače 555 a GPIO5, ke kterému je připojeno tlačítko pomocí funkce **adl_ioSubscribe**. Parametry této funkce jsou uvedeny u předchozího programu (kap. 4.2.3).
Funkce **adl_smsSubscribe** slouží k přihlášení SIM karty ke službám SMS. Parametry této funkce jsou následující:
První parametr udává jméno funkce, která je volána při každé příchozí SMS.
Druhý parametr slouží pro zjištění událostí během posílání SMS. V tomto programu není využito.
Třetí parametr nastavuje mód přenosu SMS, v tomto případě textový.
Nakonec dojde k volání funkce **Timerstart**, která spouští časovač pro měření.

```
void Timerstart(void)
{
        // nastaveni casovace pro snimani stavu tlacitka
        timer_tla=adl_tmrSubscribe(TRUE, 10, ADL_TMR_TYPE_100MS, Gpio_Tla_TimerHandler);
        // nastaveni rychlosti snimani
        timer_mereni=adl_tmrSubscribe(TRUE, 1, ADL_TMR_TYPE_TICK,
        Gpio_555_TimerHandler);
}
```

Funkce **Timerstart** slouží ke spuštění časovačů. Časovač **timer_mereni** slouží pro měření teploty a časovač **timer_tla** je určen ke zjištění stavu tlačítka. Funkce **adl_tmrSubscribe** slouží pro přihlášení a nastavení časovače. Její parametry jsou uvedeny u předchozího programu (kap. 4.4.3).

```
bool SmsHandler(ascii *SmsTel, ascii *SmsTimeOrLength, ascii *SmsText)
{
        adl_atSendResponse(ADL_AT_UNS, "Dosla SMS z telefonniho cisla: ");
        adl_atSendResponse(ADL_AT_UNS, SmsTel);
        adl_atSendResponse(ADL_AT_UNS, "\r\n");
        adl_atSendResponse(ADL_AT_UNS, "Text prichozi zpravy: ");
        adl_atSendResponse(ADL_AT_UNS, SmsText);
        adl_atSendResponse(ADL_AT_UNS, "\r\n" );
        if(wm_strcmp( "openat",SmsText ) == 0) posliSMS(SmsTel);
        return FALSE;
}
```

Funkce **SmsHandler** je handler funkce **adl_smsSubscribe** a je volaný při každé příchozí SMS. Parametr **SmsTel** obsahuje telefonní číslo odesilatele SMS, **SmsTimeOrLenght** obsahuje čas, kdy byla SMS přijata a **SmsText** obsahuje text SMS zprávy. Všechny parametry jsou v podobě textových řetězců v ASCII kódu. Pokud chceme, aby se příchozí SMS uložila na SIM kartu zvolíme návratovou hodnotu funkce return TRUE. Pokud nepožadujeme její uložení na SIM kartu, zvolíme return FALSE.
Obsah příchozí SMS zprávy je vypsán společně s číslem, ze kterého zpráva přišla do okna terminálu pomocí funkce **adl_atSendResponse**. Obdrží-li modem SMS zprávu s textem "**openat**", dojde k zavolání funkce **posliSMS** s parametrem **SmsTel**.

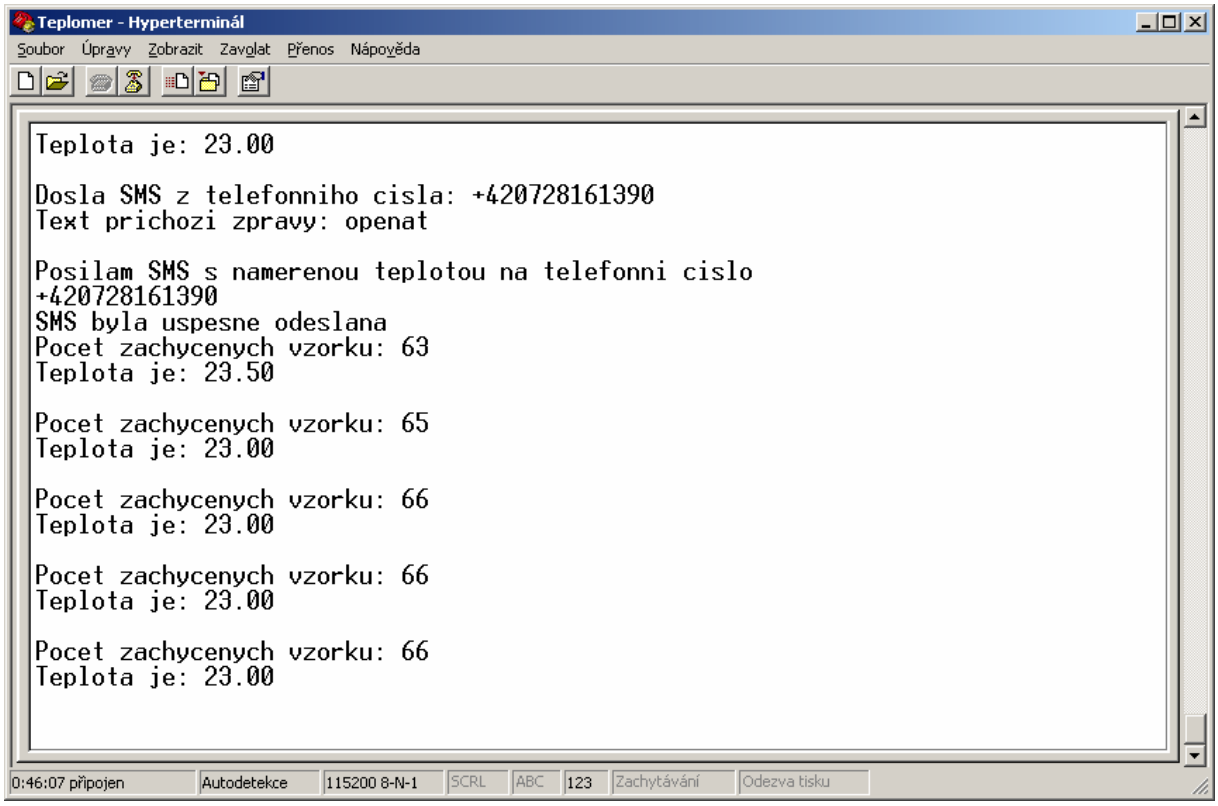
```
void Gpio_Tla_TimerHandler(u8 timerId)
{
	prectitla=adl_ioRead(stavtla);
	if (prectitla!=0) posliSMS("+420603466918");
}
```

Funkce **Gpio_Tla_TimerHandler** je volána při přetečení časovače **timer_tla**. Slouží ke zjištění stavu tlačítka pomocí funkce **adl_ioRead**, která přečte stav portu GPIO5. Bude-li tlačítko stisknuto (návratová hodnota funkce bude 0xFFFFFFFF), dojde k zavolání funkce **posliSMS** s parametrem telefonního čísla, na které bude poslána zpráva s naměřenou teplotou.

```
void posliSMS(ascii *telcislo)
{
	un_timer_mereni=adl_tmrUnSubscribe(timer_mereni, Gpio_555_TimerHandler,
	ADL_TMR_TYPE_TICK);
	un_timer_tla=adl_tmrUnSubscribe(timer_tla, Gpio_Tla_TimerHandler,
	ADL_TMR_TYPE_100MS);
	adl_atSendResponse(ADL_AT_UNS, "\r\nPosilam SMS s namerenou teplotou na telefonni cislo
	\r\n"); // vypise stav tlacitka do okna Terminalu
	adl_atSendResponse(ADL_AT_UNS, telcislo );
	strcpy(SmsText,"");
	strcat(SmsText,"Namerena teplota: ");
	strcat(SmsText,asciiteplota);
	strcat(SmsText," stupnu.");
	// zadani textu SMS a tel. cisla, na ktere se ma zprava poslat
	vysledek_sms=adl_smsSend(sms, telcislo, SmsText, ADL_SMS_MODE_TEXT);

	switch (vysledek_sms) // vypise, zda byla SMS odeslana, nebo chybu, kvuli ktere nemohla byt
		odeslana
	{
		case OK:
		adl_atSendResponse(ADL_AT_UNS,"\r\nSMS byla uspesne odeslana\r\n");
		break;

		case ADL_RET_ERR_PARAM:
		adl_atSendResponse(ADL_AT_UNS,"\r\nParametr ma spatnou hodnotu\r\n");
		break;

		case ADL_RET_ERR_UNKNOWN_HDL:
		adl_atSendResponse(ADL_AT_UNS,"\r\ntNeznamy ukazatel\r\n");
		break;

		case ADL_RET_ERR_BAD_STATE:
		adl_atSendResponse(ADL_AT_UNS,"\r\nSMS nemohla byt poslana, protoze
		neprobehla inicializace\r\n");
		break;
	}
	Timerstart();
}
```

Funkce **posliSMS** slouží k odeslání SMS zprávy z údaji o naměřené teplotě. Nejprve dojde k odhlášení časovačů pomocí funkce **adl_tmrUnSubscribe**. Sestaví se text SMS zprávy, který je uložen do proměnné **SmsText**. Správa je následně odeslána pomocí funkce **adl_smsSend**. Parametry funkce jsou následující:
První parametr udává hodnotu, kterou vrací funkce **adl_smsSubscribe.**

Druhý parametr udává telefonní číslo, na které chceme SMS poslat.
Třetí parametr udává text SMS zprávy.
Čtvrtý parametr udává mód – v tomto případě textový.
Proměnná **vysledek_sms** slouží ke kontrole, zda byla SMS odeslána v pořádku, nebo došlo k chybě, kvůli které nemohla byt odeslána.
Po odeslání zprávy dojde k zavolání funkce **Timerstart**.



Obr. 5.1: Ukázka aplikace pro měření teploty s posíláním SMS zprávy

6. Závěr

S pomocí platformy Open AT můžeme vytvářet aplikace určené k ovládání a využívání služeb bezdrátových modemů v sítí GSM a GPRS pomocí programovacího jazyka C. Vytvořené aplikace mohou být určeny například pro odesílání informací o stavu zařízení, naměřené teplotě, souřadnicích GPS nebo také sloužit ke kontrole objektů apod. Získaná data se mohou posílat jako textové SMS zprávy prostřednictvím GSM sítě, pomocí GPRS nahrát na vzdálený FTP server nebo poslat jako emailová zpráva.

V rámci bakalářské práci byl realizován jednoduchý modul pro měření teploty pracující na principu převodu měřené teploty na kmitočet. Signál obdélníkového průběhu je snímám pomocí programovatelných vstupně/výstupních pinů modemu Wavecom Fastrack M1306B s logickými úrovněmi CMOS.

Na bázi platformy Open AT byly vytvořeny ve vývojovém prostředí Microsoft Visual C++ 6.0 dvě aplikace. První aplikace odesílá údaje o naměřené teplotě, datu a času měření na FTP server prostřednictvím GPRS. Tyto údaje lze zobrazit formou tabulky a grafu na webové stránce. Druhá aplikace odesílá hodnotu naměřené teploty jako odpověď v textové SMS zprávě. Zdrojové kódy aplikací jsou detailně rozebrány a popsány tak, aby mohli sloužit jako podklady pro laboratorní úlohy věnované programování aplikací pro GSM modemy na bázi platformy Open AT.

Literatura

[1] AVX. *X5R Dielectric General Specifications* [online]. Poslední aktualizace 2006-12-04 [cit. 2007-12-09]. Ve formátu PDF. Dostupný z WWW: < http://www.avxcorp.com/docs/Catalogs/cx5r.pdf>.

[2] Epcos. *Katalogové listy termistoru K164NK100* [online]. Poslední aktualizace 2006-03-14 [cit 2007-12-12]. Ve formátu PDF. Dostupný z WWW: <http://www.epcos.com/inf/50/db/ntc_06/LeadedDisks__B57164__K164.pdf>.

[3] Google. *Google Chart API - Developer's Guide* [online]. 2008 [cit. 2008-04-25]. Dostupný z WWW: <http://code.google.com/apis/chart>.

[4] Chytil, J. *Regrese ve VB 2.díl* [online]. Poslední aktualizace 2007-03-05 [cit. 2008-04-21]. Dostupné z WWW: <http://programujte.com/index.php?akce=clanek&cl=2007030402-regrese-ve-vb-2-dil>

[5] ŠILHAVÝ, P. *Programování GSM modemů na bázi Open AT* [online]. Poslední aktualizace 2007-09-21 [cit. 2007-12-12]. Ve formátu PDF. Dostupný z WWW: <http://www.utko.feec.vutbr.cz/~silhavy/MVDP/index_soubory/3_MVDP.pdf>.

[6] Texas Instruments. *Katalogové listy časovače TLC555* [online]. Poslední aktualizace 2007-10-12 [cit. 2007-12-12]. Ve formátu PDF. Dostupný z WWW: <http://focus.ti.com/lit/ds/symlink/tlc555.pdf>.

[7] Texas Instruments. *Katalogové listy integrovaného obvodu CD40106B* [online]. Poslední aktualizace 2007-10-10 [cit. 2007-12-12]. Ve formátu PDF. Dostupný WWW: <http://focus.ti.com/lit/ds/symlink/cd40106b.pdf>.

[8] Texas Instruments. *Katalogové listy stabilizátoru napětí TL317* [online]. Poslední aktualizace 2007-10-06 [cit. 2007-12-12]. Ve formátu PDF. Dostupný z WWW: <http://focus.ti.com/lit/ds/symlink/tl317.pdf>.

[9] Wavecom. *ADL user guide for Open AT* [online]. Poslední aktualizace 2005-12-12 [cit. 2007-12-02]. Ve formátu PDF. Dostupný z WWW: <http://cst.mi.fu-berlin.de/teaching/SS06/19514-P-TI/Documentation/AUG_ADL_User_Guide_for_Open_AT_V303_rev001.pdf>.

[10] Wavecom. *AT Commands Interface Guide* [online]. Poslední aktualizace 2005-12-09 [cit. 2008-04-25]. Ve formátu PDF. Dostupný z WWW: <http://cst.mi.fu-berlin.de/teaching/SS06/19514-P-TI/Documentation/AT_Commands_Interface_Guide_for_X51a_rev006.pdf>.

[11] Wavecom. *Basic development guide for Open AT* [online]. Poslední aktualizace 2005-11-29 [cit. 2007-12-02]. Ve formátu PDF. Dostupný z WWW: <http://cst.mi.fu-berlin.de/teaching/SS06/19514-P-TI/Documentation/BUG_Basic_Development_Guide_for_Open_AT_v303_rev001.pdf>

[12] Wavecom. *Fastrack M1306B user guide* [online]. Poslední aktualizace 2006-11-11 [cit. 2007-12-02]. Ve formátu PDF. Dostupný z WWW: <http://www.linkwave.co.uk/assets/Fastrack_M1306B_User_Guide_rev003.pdf >.

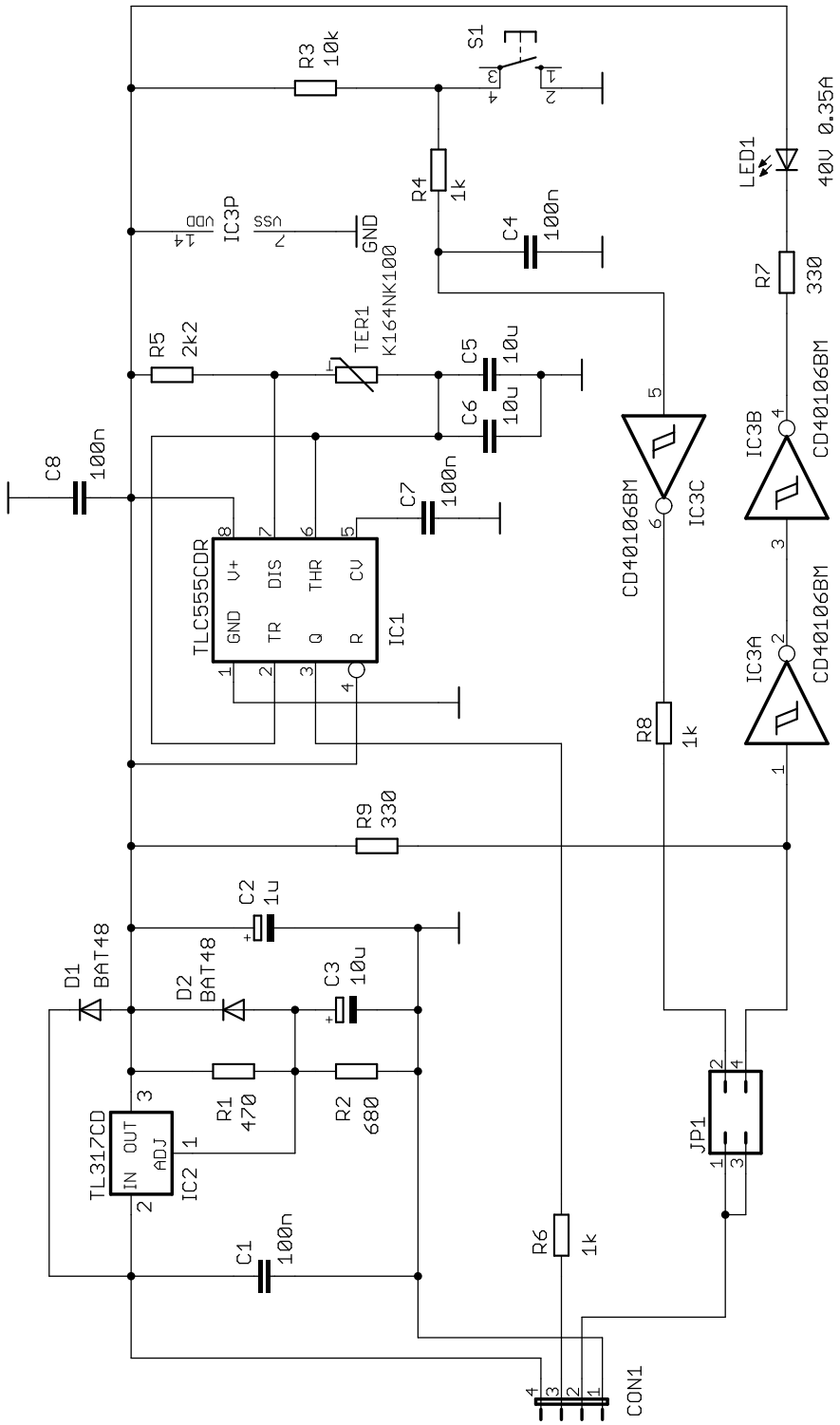
[13] Wavecom. *Open AT IP Connectivity Development Guide (eDLib V3.10)* [online]. Poslední aktualizace 2005-07-12 [cit. 2008-04-21]. Ve formátu PDF. Dostupný z WWW: <http://cst.mi.fu-berlin.de/teaching/SS08/19514-P-TI/Dokumentation/P_Open_AT_Dev_Guide_for_IP_Connectivity-004.pdf>.

[14] Zajíc, P. *Seriál PHP* [online]. Poslední aktualizace 2006-02-04 [cit. 2008-04-25]. Ve formátu PDF. Dostupný z WWW: <http://www.linuxsoft.cz/php/Serial-PHP.pdf>.

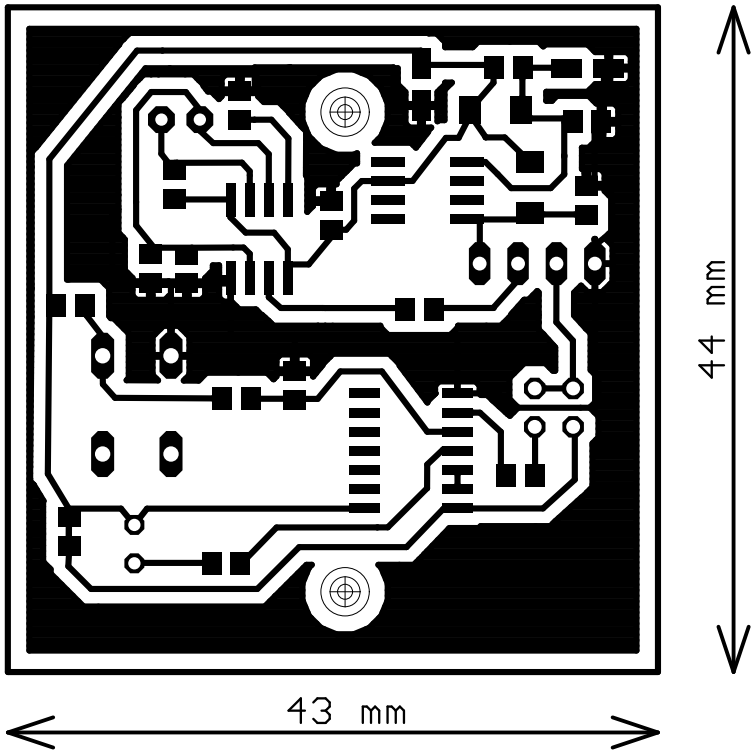
Seznam příloh

A. Výkresová dokumentace modulu pro převod teploty na frekvenci

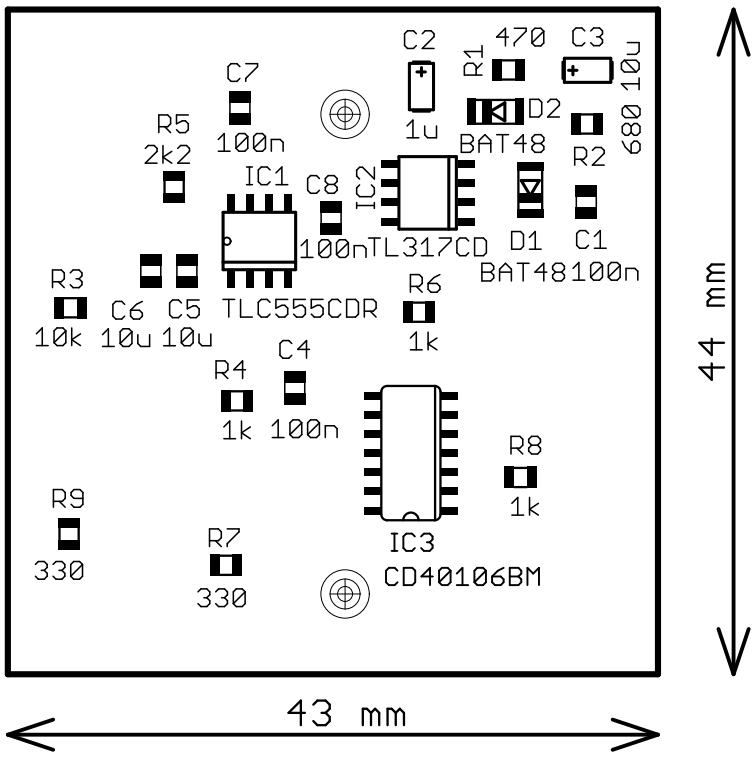
A.1. Schéma zapojení



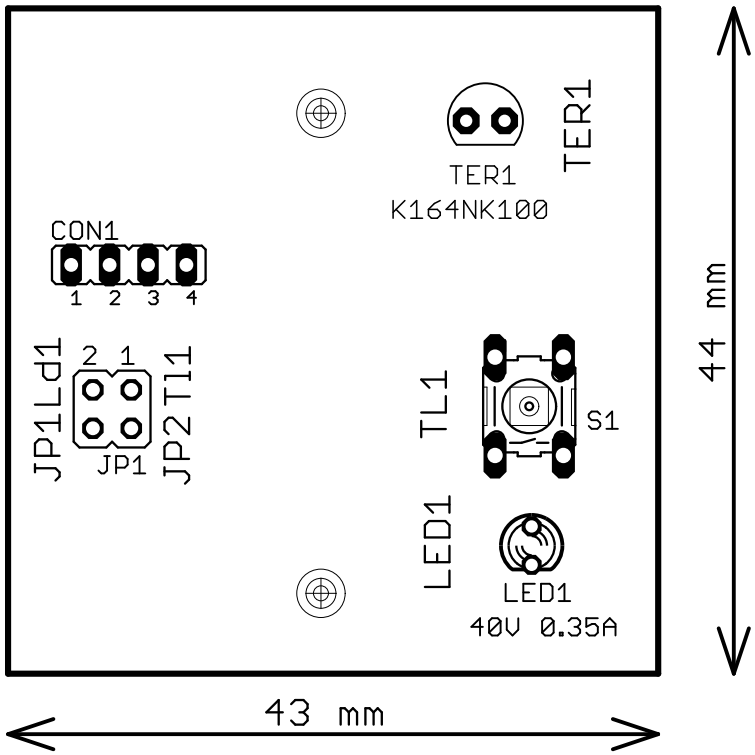
A.2. Spodní strany desky plošného spoje – matrice (2:1)



A.3. Spodní strany desky plošného spoje – rozmístění součástek (2:1)



A.4. Horní strana desky plošného spoje – rozmístění součástek (2:1)



B. Seznam součástek

Označení	Pouzdro	Hodnota/Název
C_2	Tantal. vel. A	1μF
C_3	Tantal. vel. A	10μF
C_1 C_4 C_7 C_8	805	100nF/50 V
C_5 C_6	1206	10μF
R_1	805	470Ω
R_2	805	680Ω
R_3	805	10kΩ
R_4 R_6 R_8	805	1kΩ
R_5	805	2,2kΩ
R_7 R_9	805	330Ω
D_1 D_2	Minimelf	BAT48 40V/350mA
LED_1	3mm žlutá	2.0V(max. 2.4V)/10mA,
IO_1	SO8	TLC555CDR
IO_2	SO8	TL317CD
IO_3	SO14	CD40106BM
S_1	Tlačítko 6,5 x 4,5 mm	
TER_1	K164NK100 termistor R=100Ω při 25°C	
CON_1	Konektor - vidlice 4pin, zahnutá, RM =2,54mm	
JP_1	2 x kontaktní piny, RM = 2,54mm	

C. Zdrojový kód webové stránky pro zobrazení údajů o naměřené teploty

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
 <head>
  <meta http-equiv="content-type" content="text/html; charset=windows-1250">
  <meta name="generator" content="PSPad editor, www.pspad.com">
  <title>     Open AT - Měření teploty
  </title>
 </head>
 <body bgcolor="#CCFFFF" text="black">

<?
$i = -1; //vynechaní . ..
$j = 0;  //pocet souboru
$adresar=opendir("./mereni/"); //otevreni adresare

while($jmenosouboru[$i] = readdir($adresar)) {  //nacteni nazvu souboru
$i++;
}
closedir($adresar);

for ($n=$i-1; $n>0; $n--){  //zerazeni od nejnovejsiho souboru po nejstarsi
$j++;
$soubor = fopen("./mereni/$jmenosouboru[$n]", "r");  //otevreni souboru
$teplota[$j]=fgets($soubor); //prvni radek
$datum[$j]=fgets($soubor); // druhy radek
$cas[$j]=fgets($soubor); // treti radek
fclose($soubor);    //zavreni souboru
}
?>

<table cellpadding="4" border="0" cellspacing="0" width="1000" align="center" >
<TR><TD colspan="3"><H1><center>Informace o naměřené teplotě jsou na tento server odesílány z
modemu FASTRACK M1306B přes GSM síť pomocí GPRS</center></H1></TD></TR>
<TR>
<TD rowspan="3" width="300">
<table cellpadding="0" border="0" cellspacing="5" width="250" align="center">
<TR
bgcolor="#66FF99"><TD><B>Datum</B></TD><TD><B>Čas</B></TD><TD><B>Teplota</B></T
D></TR>
<?
for ($k=1; $k<=$j; $k++){ //vypis tabulky z namerenymi teplotami; radky meni barvu
if ($k/2 != floor($k/2)) {$barva="aqua";} //liche cislo
else {$barva="#D0D0FF";}                  //sude cislo
echo "<TR
bgcolor=".$barva."><TD>".$datum[$k]."</TD><TD>".$cas[$k]."</TD><TD>".$teplota[$k]."
°C</TD></TR>\n";
}
?>
</table>
</TD>
<TD width="200" height="50" align="Center" valign="top">
<b>Vyberte datum: </b>
</TD>
<TD width="500" height="50" align="left" valign="top">
<form action="index.php"  method="get">
```

```
<SELECT NAME="den">
  <? for ($d=1; $d<=9; $d++) echo
          "<OPTION VALUE=0$d>".$d; ?>
  <? for ($d=10; $d<=31; $d++) echo
          "<OPTION VALUE=$d>".$d; ?>
</SELECT>
<SELECT NAME="mesic">
          <OPTION VALUE="01">Leden
          <OPTION VALUE="02">Únor
          <OPTION VALUE="03">Březen
          <OPTION VALUE="04">Duben
          <OPTION VALUE="05">Květen
          <OPTION VALUE="06">Červen
          <OPTION VALUE="07">Červenec
          <OPTION VALUE="08">Srepen
          <OPTION VALUE="09">Září
          <OPTION VALUE="10">Říjen
          <OPTION VALUE="11">Listopad
          <OPTION VALUE="12">Prosinec
</SELECT>
<SELECT NAME="rok">
  <? for ($r=2008; $r<=2030; $r++) echo
          "<OPTION VALUE=$r>".$r; ?>
</SELECT>
<input type="submit" value="odeslat">
</form>
</TD>
</TR>
<TR>
<TD width="700" colspan="2" valign="top" align="left">
<center><H3>Graf naměřených teplot ze dne <? echo $den.".".$mesic.".".$rok ?>
</H3></center>
<?
for ($k=1; $k<=$j; $k++){ //vymazani nepotrebnych znaku
$osetreny_datum[$k]=preg_replace('/([^A-Za-z0-9]+)/',",$datum[$k]);
}

$vyber=$den.$mesic.$rok;  //sestaveni promenne zvoleneho datumu
$pocet=1;   //pocet mereni odpovidajici zvoleneho datumu

for ($k=1; $k<=$j; $k++)  //nalezeni teploty a casu odpovidajici zvolenemu datumu
if (strcmp($vyber,$osetreny_datum[$k])==0) {
$zteplota[$pocet]=$teplota[$k]; $zcas[$pocet]=$cas[$k];$pocet++;
}

for ($k=1; $k<$pocet; $k++){
$hh[$k]=substr("$zcas[$k]", 0, 2);
$mm[$k]=substr("$zcas[$k]", 3, 4);
settype($zteplota[$k], "float");  //prevod ze stringu na float
settype($hh[$k], "integer");      //prevod ze stringu na integer
settype($mm[$k], "integer");      //prevod ze stringu na integer
$zcas[$k]=60*$hh[$k]+$mm[$k];     //prevod z casu hh:mm na sekundy
}

for ($k=1; $k<$pocet; $k++){    //procentualni vyjadreni hodnot pro graf
$gteplota[$k]=$zteplota[$k]/40*100;
$gcas[$k]=$zcas[$k]/60/24*100;
}

for ($k=1; $k<$pocet; $k++){    //sestaveni souradnic pro graf
```

```
$osax=$osax.$gcas[$k];
$osay=$osay.$gteplota[$k];
if ($k<$pocet-1) {$osax=$osax.","; $osay=$osay.",";}  //na konec neprida ,
}
?>
<img
src="http://chart.apis.google.com/chart?cht=s&chd=t:
<? echo $osax; ?>
|
<? echo $osay; ?>
&chg=8.33333,5,1,5
&chs=666x450&chxt=x,y,x,y&chxl=2:||Doba mereni [H]|3:||Teplota
['C]&chxr=0,0,24|1,0,40&chm=d,FF0000,1,1.0,10.0"
alt="Graf teploty" />
</TD>
</TR>
<TR>
<TD width="700" colspan="2" height=" <? $mezera=18*$j; echo $mezera; ?> " >
</TD>
</TR>
</table>
</body>
</html>
```

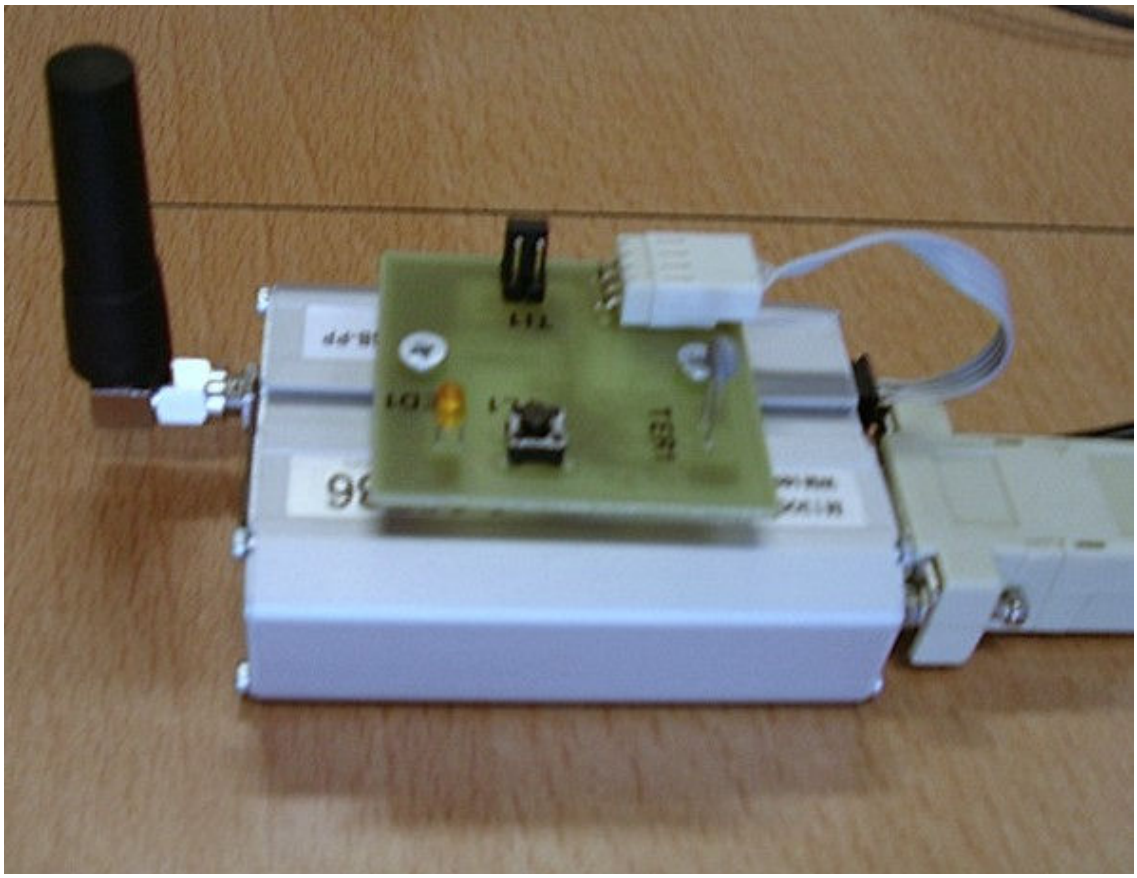
D. Funkce aplikace pro měření teploty s přenosem dat na FTP server

Název funkce	Popis
adl_main	hlavní funkce programu, provádí inicializaci SIM karty
SimHandler	handler funkce adl_simSubscribe, zjistí stav SIM karty
Timerstart	zajistí spuštění časovačů pro měření teploty a pro periodické posílání naměřené teploty na FTP server
Config	konfigurace GPRS a FTP parametrů
AtFTPtime	funkce spuštěná AT příkazem AT+FTPTIME, slouží k nastaví periodické posílání dat na FTP server
AtFTPserv	funkce spuštěná AT příkazem AT+FTPSERV, slouží k nastaví adresy FTP serveru
AtFTPun	funkce spuštěná AT příkazem AT+FTPUN, slouží k nastaví uživatelského jména pro přihlášení k FTP serveru
AtFTPpw	funkce spuštěná AT příkazem AT+FTPPW, slouží k nastavení hesla pro přihlášení k FTP serveru
AtGPRSapn	funkce spuštěná AT příkazem AT+GPRSAPN, slouží k nastaví APN pro GPRS připojení
AtGPRSun	funkce spuštěná AT příkazem AT+GPRSUN, slouží k nastaví uživatelského jména pro připojení k GPRS
AtGPRSpw	funkce spuštěná AT příkazem AT+FTPPW, slouží k nastavení hesla pro připojení k GPRS
Ftp_TimerHandler	funkce volána po vypršení časovače timer_ftp, zjistí aktuální čas
Ftp_Config_TimerHandler	sestaví název souboru a cestu na FTP serveru, kam budou data posílána, nastaví GPRS parametry pro připojení
Send_TimerHandler	funkce volána po vypršení časovače, slouží k navázání GPRS spojení
DialupHandler	handler funkce ed_DialupConnectionStart, kontroluje stav GPRS připojení
ConectFTP	provede spojení s FTP serverem a nastaví parametry určené k přenosu dat na FTP server
FtpPutHandler	handler funkce ed_FTPPut, kontroluje stavu FTP přenosu
FtpPutData	přenos dat na FTP server
Send_TimerHandler2	funkce volána po vypršení časovače, slouží k rozpojení GPRS spojení
DialupHandler2	handler funkce ed_DialupConnectionStop, kontroluje stav GPRS rozpojení
ClockHandler	sestaví datum měření do formátu “dd.mm.rr“, čas měření do formátu “hh:mm“ a jméno souboru posílaného na FTP server
Gpio_TimerHandler	funkce volána po vypršení časovače timer_mereni, čte stav portu GPIO4 a provádí výpočet měřené teploty

E. Funkce aplikace pro měření teploty s posíláním SMS zpráv

Název funkce	Popis
adl_main	hlavní funkce programu, provádí inicializaci SIM karty
SimHandler	handler funkce adl_simSubscribe, zjistí stav SIM karty
Timerstart	zajistí spuštění časovačů pro měření teploty a ke zjištění stavu tlačítka
SmsHandler	handler funkce adl_smsSubscribe, je volán při každé příchozí SMS
Gpio_Tla_TimerHandler	funkce volána při vypršení časovače timer_tla, při stisku tlačítka volá funkci posliSMS
posliSMS	odešle SMS zprávu s údaji o naměřené teplotě
Gpio_555_TimerHandler	funkce volána po vypršení časovače timer_mereni, čte stav portu GPIO4 a provádí výpočet měřené teploty

F. Modul pro měření teploty



G. Obsah přiloženého CD

Složka / soubor	Popis
Binární soubory	složka obsahuje zkompilované aplikace určené pro přímé nahrání a spuštění na modemu
gcc_teplomer_32.wpb.dwl	aplikace pro měření teploty s přenosem dat na FTP server
gcc_SMS_32.wpb.dwl	aplikace pro měření teploty s posíláním SMS zpráv
Katalogové listy	složka obsahuje katalogové listy použitých součástek pro sestavení modulu pro měření teploty
cd40106b.pdf	katalogové listy integrovaného obvodu CD40106B
LeadedDisks__B57164__K164.pdf	katalogové listy NTC termistorů firmy Epcos
tl317.pdf	katalogové listy stabilizátoru napětí TL317
tlc555.pdf	katalogové listy časovače TLC555
Manuály	složka obsahuje manuály k modemu Fastrack M1306B a platformě Open AT
Open AT aplikace	složka obsahuje Open AT aplikace se zdrojovými kódy pro vývojové prostředí Microsoft Visual C++ 6.0
Teplomer	aplikace pro měření teploty s přenosem dat na FTP server se zdrojovými kódy
SMS	aplikace pro měření teploty s posíláním SMS zpráv se zdrojovými kódy
Schéma a DPS	složka obsahuje schéma a návrh desky plošného spoje modulu pro měření teploty v programu Eagle
WWW	složka obsahuje zdrojový kód webové stránky pro zobrazení naměřených teplot v internetovém prohlížeči
Bakalářská práce.pdf	bakalářská práce
Metadata.pdf	popisný soubor bakalářské práce